File 1: igw\com\Application.as

```
package com
{
import com.service.ExternalInterfaceAPI;

import flash.display.DisplayObject;
import flash.display.LoaderInfo;
import flash.display.Stage;
import flash.events.ErrorEvent;
import flash.events.UncaughtErrorEvent;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.osflash.signals.Signal;
import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;

public class Application
{
// Network identifiers -- what platform are we running on?
// Values here must match the "play_platform_id" used by the Kabam platform.
// On the server the enumeration is KabamPlayPlatformId.
// Note this document too for reference:
https://sites.google.com/a/watercooler-inc.com/kabam-site/kfid
// ...however ignore the example id's listed there, as they differ from the play platform id.
public static const NETWORK_UNKNOWN:int = -1;
public static const NETWORK_FACEBOOK:int = 0;
public static const NETWORK_KABAM:int = 1;
//public static const NETWORK_IOS:int = 2;
//public static const NETWORK_GOOGLE:int = 3; // Possibly deprecated...?
//public static const NETWORK_ANDROID:int = 4; // Apparently deprecated (use
NETWORK_GOOGLEAPP).
public static const NETWORK_YAHOO:int = 12;
public static const NETWORK_KONGREGATE:int = 36;
public static const NETWORK_STEAM:int = 48;
public static const NETWORK_DEV:int = 49;
public static const NETWORK_GOOGLEAPP:int = 51;
//public static const NETWORK_AMAZONAPP:int = 52;
public static const NETWORK_XSOLLA:int = 53;
public static const NETWORK_GUEST:int = 54;

public static var ASSET_PATH:String = "";
public static var AVG_LOAD_TIME:int;
public static var CONNECTION_STATE:String;
public static var MIN_SCREEN_X:Number = 1280;
public static var MIN_SCREEN_Y:Number = 800;
public static var NETWORK:int = NETWORK_UNKNOWN;
public static var PLAYER_KEY:String;
public static var PLAYER_KABAM_NAID:String;
public static var LANGUAGE:String;
public
```

```
static var COUNTRY:String;
public static var PLAYER_TOKEN:String;
public static var PLAYER_OAUTH:String;
public static var PROXY_PORT:int;
public static var PROXY_SERVER:String;
public static var SCALE:Number = .7;
public static var STAGE:Stage;
public static var STARLING_ENABLED:Boolean;
public static var STATE:String;
public static var LOGIN_TOKEN:String;
public static var BATTLE_WEB_PATH:String;

public static var batteryLife:Number;
public static var isCharging:Boolean;

public static var onBatteryChargeChanged:Signal;
public static var onError:Signal;


private static var _root:DisplayObject;

private static const _logger:ILogger = getLogger('Application');

public static function init( stage:Stage ):void
{
//ExternalInterfaceAPI.logConsole("Imperium Init");

STAGE = stage;
AVG_LOAD_TIME = 0;

onBatteryChargeChanged = new Signal(Boolean, Number);
onError = new Signal(String);

var data:Object = rootParameters;
if (data)
{
//TODO hack for quick test
//NETWORK = NETWORK_XSOLLA; //
//ExternalInterfaceAPI.logConsole("Imperium Platform = "+ data["play-platform"]);

NETWORK = data["play-platform"] ? data["play-platform"] : NETWORK_XSOLLA;
LOGIN_TOKEN = data["login-token"];
ExternalInterfaceAPI.setPlayPlatform(NETWORK);

ExternalInterfaceAPI.submitKongregateStat("initialized", 1);
}
BATTLE_WEB_PATH = ExternalInterfaceAPI.getBattleWebPath();
trace(DeviceMetrics.toString());
}

private
```

```
static function uncaughtErrorHandler( event:UncaughtErrorEvent ):void
{
event.preventDefault();

var errStr:String;
if (event.error is Error)
{
var error:Error = event.error as Error;
errStr = error.getStackTrace();
} else if (event.error is ErrorEvent)
{
var errorEvent:ErrorEvent = event.error as ErrorEvent;
errStr = errorEvent.toString();
// do something with the error
} else
{
// a non-Error, non-ErrorEvent type was thrown and uncaught
}

onError.dispatch(errStr);
}

public static function set ROOT( v:DisplayObject ):void
{
_root = v;

_root.loaderInfo.uncaughtErrorEvents.addEventListener(UncaughtErrorEvent.UNCAUGHT_ERROR,
uncaughtErrorHandler, false, 999999);
}


public static function get rootParameters():Object
{
var obj:Object = STAGE.stage.getChildAt(0); //get the parent if this swf was loaded in
var data:Object = (obj is Imperium || DeviceMetrics.PLATFORM == PlatformEnum.MOBILE) ?
LoaderInfo(STAGE.root.loaderInfo).parameters : obj.getParameters();

return data;
}

public static function setBatteryStatus( charging:Boolean, chargeRatio:Number ):void
{
isCharging = charging;
batteryLife = chargeRatio;
onBatteryChargeChanged.dispatch(charging, chargeRatio);
_logger.info("Battery status changed: isCharging = " + isCharging + ", batteryLife = " +
batteryLife + ".");
}
}
}

-----------------------------------
```

File 2: igw\com\StartupBaseConfig.as

```
package com
{
import com.controller.command.state.StartupCoreCommand;
import com.event.StateEvent;

import flash.events.IEventDispatcher;

import org.robotlegs.extensions.eventCommandMap.api.IEventCommandMap;

public class StartupBaseConfig
{
public function StartupBaseConfig( commandMap:IEventCommandMap,
dispatcher:IEventDispatcher )
{
commandMap.map(StateEvent.STARTUP_COMPLETE,
StateEvent).toCommand(StartupCoreCommand);
}

protected function startupComplete( dispatcher:IEventDispatcher ):void
{
//send out the state event
dispatcher.dispatchEvent(new StateEvent(StateEvent.STARTUP_COMPLETE));
}
}
}
```

File 3: igw\com\StartupConfig.as

```
package com
{
import com.controller.command.state.StartupCommand;
import com.event.StateEvent;

import flash.display.DisplayObjectContainer;
import flash.events.IEventDispatcher;

import org.parade.util.DeviceMetrics;
import org.robotlegs.extensions.eventCommandMap.api.IEventCommandMap;

import com.service.ExternalInterfaceAPI;

public class StartupConfig extends StartupBaseConfig
{
private static var _FontArray:Array;

public function StartupConfig( commandMap:IEventCommandMap,
dispatcher:IEventDispatcher, contextView:DisplayObjectContainer )
{

super(commandMap,
```

```
dispatcher);

commandMap.map(StateEvent.STARTUP_COMPLETE,
StateEvent).toCommand(StartupCommand);

DeviceMetrics.init(contextView.stage, CONFIG::PLATFORM);

_FontArray = new Array();
_FontArray.push(ExternalInterfaceAPI.getFont(0));
_FontArray.push(ExternalInterfaceAPI.getFont(1));
Application.init(contextView.stage);

startupComplete(dispatcher);
}
public static function get FontArray():Array
{
return _FontArray;
}

}
}
```

File 4: igw\com\controller\ChatController.as

```
package com.controller
{
import com.enum.PlayerUpdateEnum;
import com.enum.server.ChatChannelEnum;
import com.enum.server.ChatResponseCodeEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.game.entity.systems.shared.background.BackgroundSystem;
import com.model.chat.ChatChannelVO;
import com.model.chat.ChatModel;
import com.model.chat.ChatPanelVO;
import com.model.chat.ChatVO;
import com.model.player.CurrentUser;
import com.model.sector.SectorModel;
import com.service.language.Localization;
import com.service.server.incoming.chat.ChatBaselineResponse;
import com.service.server.incoming.chat.ChatResponse;
import com.service.server.outgoing.chat.ChatChangeRoomRequest;
import com.service.server.outgoing.chat.ChatIgnoreChatRequest;
import com.service.server.outgoing.chat.ChatReportChatRequest;
import com.service.server.outgoing.chat.ChatSendChatRequest;

import flash.utils.Dictionary;

import org.adobe.utils.StringUtil;
import org.as3commons.logging.api.ILogger;
import
```

```
org.as3commons.logging.api.getLogger;
import org.ash.core.Game;
import org.shared.ObjectPool;

public class ChatController
{
public var chatHasFocus:Boolean = false;

private var _chatModel:ChatModel;
private var _slashCommands:Dictionary;

private var _sectorModel:SectorModel;
private var _serverController:ServerController;
private var _backgroundSystem:BackgroundSystem
private var _game:Game;

private var _printResults:Boolean;
private var _loggedIn:Boolean;
private var _lostConnection:Boolean;

private var _sector:String = 'CodeString.Chat.ChannelName.Sector'; //Sector
private var _group:String = 'CodeString.Chat.ChannelName.Group'; //Group
private var _faction:String = 'CodeString.Chat.ChannelName.Faction'; //Faction
private var _alliance:String = 'CodeString.Chat.ChannelName.Alliance'; //Alliance
private var _members:String = 'CodeString.Chat.ChannelName.Members'; //Members
private var _centerspace:String = 'CodeString.Chat.ChannelName.CenterSpace'; //Center
Space
private var _listBlockedUsers:String = 'CodeString.Chat.SlashCommand.ListBlockedUsers';
//listblockedusers
private var _getChannelUsers:String = 'CodeString.Chat.SlashCommand.GetChannelUsers';
//getchannelusers
private var _help:String = 'CodeString.Chat.SlashCommand.Help'; //help
private var _helpText:String = 'CodeString.Chat.Message.HelpText'; //Valid Slash
Commands:<br>/s - Sector Chat<br>/g - Group Chat<br>/f - Faction Chat<br>/a - Alliance
Chat<br>/c - Center Space Chat<br>/getchannelusers - Gets a list of current channel users<br>
/listBlockedUsers - Gets a list of blocked users
private var _blockedUser:String = 'CodeString.Chat.Message.BlockedUser'; //Blocked User
private var _unblockedUser:String = 'CodeString.Chat.Message.UnblockedUser'; //Unblocked
User
private var _blockedUsers:String = 'CodeString.Chat.Message.BlockedUsers'; //Blocked Users:
private var _channelUsers:String = 'CodeString.Chat.Message.ChannelUsers'; //Channel Users:

private var _sectorChannelID:String = 'sector';
private var _groupChannelID:String = 'group';
private var _factionChannelID:String = 'faction';
private var _allianceChannelID:String = 'alliance';
private var _membersChannelID:String = 'members';
private var _centerspaceChannelID:String = 'centerspace';
private var _systemMessageChannelID:String = 'systemmessage';

private
```

```
const _logger:ILogger = getLogger('ChatController');

[PostConstruct]
public function init():void
{
_slashCommands = new Dictionary();
_slashCommands['/s'] = ChatChannelEnum.SECTOR;
//_slashCommands['/g'] = ChatChannelEnum.GROUP;
_slashCommands['/g'] = ChatChannelEnum.GLOBAL;
_slashCommands['/a'] = ChatChannelEnum.ALLIANCE;
_slashCommands['/f'] = ChatChannelEnum.FACTION;
_slashCommands['/m'] = ChatChannelEnum.MEMBERS;
//_slashCommands['/c'] = _centerspaceChannelID;

_backgroundSystem = BackgroundSystem(_game.getSystem(BackgroundSystem));
_sectorModel.addSectorChangeListener(joinSectorChat);
CurrentUser.onPlayerUpdate.add(onPlayerUpdate);
}

public function initStrings():void
{
var locManager:Localization = Localization.instance;
_sector = locManager.getString(_sector);
_group = locManager.getString(_group);
_faction = locManager.getString(_faction);
_alliance = locManager.getString(_alliance);
_members = locManager.getString(_members);
_centerspace = locManager.getString(_centerspace).split(' ').join('');
_getChannelUsers = locManager.getString(_getChannelUsers);
_listBlockedUsers = locManager.getString(_listBlockedUsers);
_help = locManager.getString(_help);
_helpText = locManager.getString(_helpText);
_blockedUser = locManager.getString(_blockedUser);
_unblockedUser = locManager.getString(_unblockedUser);
_blockedUsers = locManager.getString(_blockedUsers);
_channelUsers = locManager.getString(_channelUsers);
}

public function give( serverController:ServerController ):void
{
_serverController = serverController;
}

private function onPlayerUpdate( updateType:int, oldValue:String, newValue:String ):void
{
if (PlayerUpdateEnum.TYPE_ALLIANCE)
{
if (oldValue != newValue)
{
if (newValue != '')
_chatModel.activateChannel(ChatChannelEnum.ALLIANCE);
```

```
else
_chatModel.deactivateChannel(ChatChannelEnum.ALLIANCE);
}
}
}

public function sendChatMessageWithDefaultChannel( message:String ):void
{
if (_chatModel.defaultChannel == null)
{
_chatModel.overrideDefaultChannelByChannelID(ChatChannelEnum.SECTOR);
_logger.error('Whoa there guy no default channel trying to set the default to sector.')
}

if (_chatModel.defaultChannel == null)
_logger.fatal('YOU HAVE NO SECTOR CHANNEL SOMETHING IS MESSED UP! GAME
OVER MAN GAME OVER')
else
sendChannelMessage(_chatModel.defaultChannel.channelID, message);
}

public function handleSlashCommand( message:String ):void
{
var messageParts:Array = message.split(' ');
var firstWord:String = messageParts[0];
var command:String;
if (firstWord.length < 3)
command = firstWord.toLowerCase();
else
{
command = firstWord.slice(1).toLowerCase();
}

switch (command)
{
case _getChannelUsers:
_printResults = true;
break;
case _listBlockedUsers:
_printResults = true;
break;
case _help:
addSystemMessage(_helpText);
break;
default:
var channelID:int = -1;
if (command.length < 3)
{
channelID = _slashCommands[command];
}
```

```
else
{
_chatModel.getChannelIdFromName(command);
}

if (channelID != -1 && _chatModel.isInChannel(channelID))
{
messageParts.shift();
sendChannelMessage(channelID, messageParts.join(' '));
}
break;
}
}

public function linkCoords( x:int, y:int ):void
{
var message:String = x + ',' + y;
if (_chatModel.defaultChannel)
sendChannelMessage(_chatModel.defaultChannel.channelID, message);
}

private function addMessage( channel:int, userNID:String, userDisplayName:String,
userFaction:String, message:String ):void
{

var chatChannel:ChatChannelVO = _chatModel.getChannelFromId(channel);
userDisplayName = StringUtil.htmlEncode(userDisplayName);
message = StringUtil.escapeHTML(message);
message = checkForCoords(message, true);
message = checkForAlliance(message);
var newMessage:ChatVO = ObjectPool.get(ChatVO);
newMessage.init(userNID, userDisplayName, userFaction, chatChannel, message);
_chatModel.addChatLog(newMessage);
}

public function addSystemMessage( message:String, faction:String = '', systemName:String = '
):void
{
var chatChannel:ChatChannelVO =
_chatModel.getChannelFromId(ChatChannelEnum.SYSTEM);
var newMessage:ChatVO = ObjectPool.get(ChatVO);
newMessage.init('none', systemName, faction, chatChannel, message);
_chatModel.addChatLog(newMessage);
}

private function checkForAlliance( message:String ):String
{

if (message.indexOf('[alliance.') != -1)
{
//
```

```
matches strings between square brackets and with alliance in it [alliance.*], ignores nested
combinations and extracts only the inner part
var allianceInSquareBracketsRegExp:RegExp = /\[alliance.[^\[\]]*?\]/g;
var messageParts:Array = message.match(allianceInSquareBracketsRegExp);
var allianceLinks:Dictionary = new Dictionary();
var len:uint = messageParts.length;
var currentPartOfMessage:String;
var indexOfDot:int;
for (var i:uint = 0; i < len; ++i)
{
currentPartOfMessage = messageParts[i];

var allianceName:String = currentPartOfMessage.replace("[alliance.", "");
allianceName = allianceName.replace("]", "");

if(allianceName != "")
{
// Prepare extracted alliance key to match database key
var allianceNameLink:String = allianceName.toLowerCase();
allianceNameLink = allianceNameLink.replace(" ", "_");
allianceLinks[currentPartOfMessage] = "<font color='#2ecc71'><a href='event:AllianceLink." +
allianceNameLink + "'>" + allianceName + "</a></font>";
}
}

// Prepare message with hyperlinks
for ( var key:String in allianceLinks )
{
var currentLink:String = allianceLinks[key] as String;
var previousReplacedIndex:int = message.indexOf(key);

while (previousReplacedIndex != -1)
{
message = message.replace(key, currentLink);
previousReplacedIndex = message.indexOf(key, currentLink.length + previousReplacedIndex);
}
}
}
return message;
}

private function checkForCoords( message:String, recieved:Boolean = false ):String
{

if (message.indexOf(',') != -1)
{
var sectorModel:SectorModel;
if (_backgroundSystem == null)
_backgroundSystem = BackgroundSystem(_game.getSystem(BackgroundSystem));

if
```

```
(_backgroundSystem)
sectorModel = _backgroundSystem.sectorModel;

var maxXCoord:Number = (sectorModel) ? sectorModel.width * 0.01 : 350;
var maxYCoord:Number = (sectorModel) ? sectorModel.height * 0.01 : 350;
var messageParts:Array = message.split(' ');
var len:uint = messageParts.length;
var currentPartOfMessage:String;
var indexOfComma:uint;
var lenOfMessagePart:uint;
var firstSetOfCoords:Boolean;
var secondSetOfCoords:Boolean;
for (var i:uint = 0; i < len; ++i)
{
currentPartOfMessage = messageParts[i];
lenOfMessagePart = currentPartOfMessage.length;
indexOfComma = currentPartOfMessage.indexOf(',');
if (indexOfComma != -1 && lenOfMessagePart > indexOfComma + 1)
{
var parenIndex:int = currentPartOfMessage.indexOf('(');
var coords1:Number = Number(currentPartOfMessage.substring(0, indexOfComma));
var coords2:Number = Number(currentPartOfMessage.substring(indexOfComma + 1,
(parenIndex != -1 && recieved) ? parenIndex : lenOfMessagePart));
firstSetOfCoords = !isNaN(coords1);
secondSetOfCoords = !isNaN(coords2);
//check if they are coords and in coords range

if (firstSetOfCoords == true && (coords1 <= maxXCoord && coords1 >= 0) &&
secondSetOfCoords == true && (coords2 <= maxYCoord && coords2 >= 0))
{
if (recieved)
{
var sector:String = currentPartOfMessage.substring(parenIndex + 1,
currentPartOfMessage.length - 1);
var linkMessage:String = coords1 + ',' + coords2 + ',' + sector;
messageParts[i] = "<font color='#a9dcff'><a href='event:CoordLink." + linkMessage + "'>" +
currentPartOfMessage + "</a></font>"
} else
{
messageParts[i] += '(' + _sectorModel.sectorID + ')';
}
}
}
}
message = messageParts.join(' ');
}
return message;
}

public function isBlocked( id:String ):Boolean
{
```

```
return _chatModel.isBlocked(id);
}

public function isMuted( id:String ):Boolean
{
return _chatModel.isMuted(id);
}

public function mutePlayer( id:String ):void
{
_chatModel.mutePlayer(id);
}

public function recievedMessage( response:ChatResponse ):void
{
switch (response.responseCode)
{
case ChatResponseCodeEnum.OK:
addMessage(response.channel, response.senderKey, response.senderName,
response.senderFaction, response.message);
break;
case ChatResponseCodeEnum.ROOM_JOINED:
_logger.info('SUCCESS: Room Joined {}', response.channel);
_chatModel.activateChannel(response.channel);
break;
case ChatResponseCodeEnum.ROOM_LEFT:
_logger.info('SUCCESS: Room Left {}', response.channel);
_chatModel.deactivateChannel(response.channel);
break;
case ChatResponseCodeEnum.NO_SUCH_PLAYER:
_logger.error('ERROR: No Such Player');
break;
case ChatResponseCodeEnum.NO_SUCH_CHAT_ROOM:
_logger.error('ERROR: No Chat Channel');
break;
}
}

private function getAggregatedChannelID( requestedChannelID:int ):String
{
var faction:int = requestedChannelID % 3;
if (faction == 0)
faction = 3;

var grouping:int = Math.floor((requestedChannelID - 1) / 9);

return faction + "." + grouping;
}

private
```

```
function sendChannelMessage( channelId:int, message:String ):void
{
message = checkForCoords(message);
var msg:ChatSendChatRequest =
ChatSendChatRequest(_serverController.getRequest(ProtocolEnum.CHAT_CLIENT,
RequestEnum.CHAT_SEND_CHAT));
msg.channel = channelId;
msg.message = message;
msg.playerKey = CurrentUser.id;
_serverController.send(msg);
}

public function sendUnblockOrBlock( userNAID:String ):void
{
_chatModel.addOrRemoveBlocked(userNAID);
var msg:ChatIgnoreChatRequest =
ChatIgnoreChatRequest(_serverController.getRequest(ProtocolEnum.CHAT_CLIENT,
RequestEnum.CHAT_IGNORE_CHAT));
msg.playerKey = userNAID;
_serverController.send(msg);
}

public function sendReportChat( userNAID:String ):void
{
var msg:ChatReportChatRequest =
ChatReportChatRequest(_serverController.getRequest(ProtocolEnum.CHAT_CLIENT,
RequestEnum.CHAT_REPORT_CHAT));
msg.playerKey = userNAID;
_serverController.send(msg);
}

public function addBlockedUsers( response:ChatBaselineResponse ):void
{
_chatModel.blockedList = response.ignoredPlayers;
}

private function joinSectorChat( id:String ):void
{
var key:int = id ? int(id.split(".")[1]) : 1;
var newID:String = "sector." + getAggregatedChannelID(key);

var msg:ChatChangeRoomRequest =
ChatChangeRoomRequest(_serverController.getRequest(ProtocolEnum.CHAT_CLIENT,
RequestEnum.CHAT_CHANGE_ROOM));
msg.roomKey = id;
msg.roomKey = newID;
msg.channel = ChatChannelEnum.SECTOR;
_serverController.send(msg);

_logger.info('Change Room Request: Sector Channel {}', id);
}
```

```
public function addChatListener( callback:Function ):void { _chatModel.chatSignal.add(callback);
}
public function removeChatListener( callback:Function ):void {
_chatModel.chatSignal.remove(callback); }

public function addOnActiveChannelUpdatedListener( callback:Function ):void {
_chatModel.activeChannelUpdated.add(callback); }
public function removeOnActiveChannelUpdatedListener( callback:Function ):void {
_chatModel.activeChannelUpdated.remove(callback); }

public function addOnDefaultChannelLoadedListener( callback:Function ):void {
_chatModel.onDefaultChannelOverriden.add(callback); }
public function removeOnDefaultChannelLoadedListener( callback:Function ):void {
_chatModel.onDefaultChannelOverriden.remove(callback); }

public function addOnDefaultChannelUpdatedListener( callback:Function ):void {
_chatModel.onDefaultChannelUpdated.add(callback); }
public function removeOnDefaultChannelUpdatedListener( callback:Function ):void {
_chatModel.onDefaultChannelUpdated.remove(callback); }

public function getDefaultChannel():ChatChannelVO { return _chatModel.defaultChannel; }
public function setDefaultChannel( newDefaultId:ChatChannelVO ):void
{
_chatModel.defaultChannel = newDefaultId;
}

public function getChannelColorFromId( channelID:int ):uint
{
return _chatModel.getChannelColorFromId(channelID);
}

public function getActiveChannels():Dictionary { return _chatModel.activeChannels; }
public function getChatPanels():Vector.<ChatPanelVO> { return _chatModel.chatPanels; }
public function getPanelLogs( panelID:uint ):String { return _chatModel.getPanelLogs(panelID); }

public function get blockedList():Vector.<String> { return _chatModel.blockedList; }

[Inject]
public function set chatModel( v:ChatModel ):void { _chatModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
}
}

-----------------------------------
File 5: igw\com\controller\EventController.as
package
```

```
com.controller
{
import com.enum.EventStateEnum;
import com.model.event.EventModel;
import com.model.event.EventVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuffVO;
import com.model.starbase.StarbaseModel;
import com.service.server.incoming.data.BuffData;

import flash.events.TimerEvent;
import flash.utils.Timer;

import org.shared.ObjectPool;

public class EventController
{
private var _eventModel:EventModel;
private var _starbaseModel:StarbaseModel;
private var _prototypeModel:PrototypeModel;

private var _timer:Timer;

private var _currentRunningEvent:EventVO;

public function EventController()
{
_timer = new Timer(1000);
_timer.addEventListener(TimerEvent.TIMER, onUpdateEvents, false, 0, true);
}

public function addEvents( active:Vector.<IPrototype>, upcoming:Vector.<IPrototype>,
now:Number ):void
{
var i:uint;
var len:uint;

var activeEvents:Vector.<EventVO> = new Vector.<EventVO>;
var upcomingEvents:Vector.<EventVO> = new Vector.<EventVO>;
var currentEvent:EventVO;
var currentProto:IPrototype;
var currentActiveEvent:EventVO;
var currentActiveEventHolder:EventVO;

len = active.length;
for (i = 0; i < len; ++i)
{
currentProto = active[i];
currentEvent
```

```
= new EventVO(currentProto, EventStateEnum.RUNNING, currentProto.getValue('eventEnds') -
now);

if (currentEvent.isUiTracking && (currentActiveEventHolder == null || (currentActiveEventHolder
&& currentEvent.timeRemainingMS < currentActiveEventHolder.timeRemainingMS)))
currentActiveEventHolder = currentEvent;

addBuffs(currentEvent);
activeEvents.push(currentEvent);
}

len = upcoming.length;
for (i = 0; i < len; ++i)
{
currentProto = upcoming[i];
currentEvent = new EventVO(currentProto, EventStateEnum.UPCOMING,
currentProto.getValue('eventBegins') - now);

if (currentActiveEvent == null && currentEvent.isUiTracking && currentEvent.timeRemainingMS
< 345600000 && (currentActiveEventHolder == null || (currentActiveEventHolder &&
currentEvent.
timeRemainingMS <
currentActiveEventHolder.timeRemainingMS)))
currentActiveEventHolder = currentEvent;

upcomingEvents.push(currentEvent);
}

currentActiveEvent = currentActiveEventHolder;

_eventModel.addEvents(currentActiveEvent, activeEvents, upcomingEvents);
_timer.start();
}



private function onUpdateEvents( e:TimerEvent ):void
{
var activeEvents:Vector.<EventVO> = _eventModel.activeEvents;
var upcomingEvents:Vector.<EventVO> = _eventModel.upcomingEvents;

var i:uint;
var len:uint;
var currentEvent:EventVO;
var currentActiveEvent:EventVO;
var currentActiveEventHolder:EventVO;

var updated:Boolean;

len = activeEvents.length;
for
```

```
(i = 0; i < len; ++i)
{
currentEvent = activeEvents[i];
if (currentEvent.timeRemainingMS <= 0)
{
updated = true;
currentEvent.state = EventStateEnum.ENDED;
removeBuffs(currentEvent);
activeEvents.splice(i, 1);
--i;
--len;
} else
{
if (currentEvent.isUiTracking && (currentActiveEventHolder == null || (currentActiveEventHolder
&& currentEvent.timeRemainingMS < currentActiveEventHolder.timeRemainingMS)))
currentActiveEventHolder = currentEvent;
}
}

len = upcomingEvents.length;
for (i = 0; i < len; ++i)
{
currentEvent = upcomingEvents[i];
if (currentEvent.timeRemainingMS <= 0)
{
updated = true;
currentEvent.state = EventStateEnum.RUNNING;
currentEvent.timeRemainingMS = currentEvent.ends - currentEvent.begins;
addBuffs(currentEvent);
upcomingEvents.splice(i, 1);
--i;
--len;
activeEvents.push(currentEvent);
} else
{
if (currentActiveEvent == null && currentEvent.isUiTracking && currentEvent.timeRemainingMS
< 345600000 && (currentActiveEventHolder == null || (currentActiveEventHolder &&
currentEvent.
timeRemainingMS <
currentActiveEventHolder.timeRemainingMS)))
currentActiveEventHolder = currentEvent;
}
}

currentActiveEvent = currentActiveEventHolder;

if (updated)
_eventModel.addEvents(currentActiveEvent, activeEvents, upcomingEvents, true);
}

public
```

```
function addBuffs( v:EventVO ):void
{
var buffs:Array = v.buffsGranted;
var len:uint = buffs.length;
var buffData:BuffData;
for (var i:uint = 0; i < len; ++i)
{
buffData = ObjectPool.get(BuffData);
buffData.baseID = _starbaseModel.currentBaseID;
buffData.began = v.begins;
buffData.ends = v.ends;
buffData.id = buffs[i];
buffData.playerOwnerID = CurrentUser.id;
buffData.prototype = _prototypeModel.getBuffPrototype(buffs[i]);
buffData.timeRemaining = v.timeRemainingMS;
_starbaseModel.importBuffData(buffData);
}


}

public function removeBuffs( v:EventVO ):void
{
var buffs:Array = v.buffsGranted;
var len:uint = buffs.length;
var currentBuff:String;
for (var i:uint = 0; i < len; ++i)
{
currentBuff = buffs[i];
_starbaseModel.removeBuffByID(currentBuff);
}
}

[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set eventModel( v:EventModel ):void { _eventModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
}
}

-----------------------------------
File 6: igw\com\controller\GameController.as
package com.controller
{
import com.Application;
import com.controller.fte.FTEController;
import com.controller.sound.SoundController;
import com.controller.transaction.TransactionController;
import
```

```
com.service.ExternalInterfaceAPI;
import com.enum.AudioEnum;
import com.enum.BattleLogFilterEnum;
import com.enum.CategoryEnum;
import com.enum.FleetStateEnum;
import com.enum.MissionEnum;
import com.enum.RemoveReasonEnum;
import com.enum.TimeLogEnum;
import com.enum.ToastEnum;
import com.enum.TypeEnum;
import com.enum.server.BattleEntityTypeEnum;
import com.enum.server.OrderEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.enum.server.SectorEntityStateEnum;
import com.enum.server.SectorEntityTypeEnum;
import com.event.BattleEvent;
import com.event.PaywallEvent;
import com.event.SectorEvent;
import com.event.ServerEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.ToastEvent;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.battle.Beam;
import com.game.entity.components.battle.DebuffTray;
import com.game.entity.components.battle.Drone;
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Modules;
import com.game.entity.components.battle.Shield;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.sector.Transgate;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Cargo;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Pylon;
import com.game.entity.components.shared.VCList;
import com.game.entity.factory.IAttackFactory;
import com.game.entity.factory.ISectorFactory;
import com.game.entity.factory.IShipFactory;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.systems.battle.DebugLineSystem;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.game.entity.systems.shared.VCSystem;
import com.game.entity.systems.shared.grid.GridSystem;
import
```

```
com.game.entity.systems.starbase.StarbaseSystem;
import com.model.achievements.AchievementModel;
import com.model.alliance.AllianceModel;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.battle.BattleModel;
import com.model.battlelog.BattleLogModel;
import com.model.blueprint.BlueprintModel;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.leaderboards.LeaderboardModel;
import com.model.mail.MailModel;
import com.model.mission.MissionModel;
import com.model.mission.MissionVO;
import com.model.motd.MotDDailyRewardModel;
import com.model.motd.MotDModel;
import com.model.player.BookmarkVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.model.starbase.StarbaseModel;
import com.model.warfrontModel.WarfrontModel;
import com.presenter.sector.ISectorPresenter;
import com.presenter.shared.IGamePresenter;
import com.presenter.starbase.IStarbasePresenter;
import com.service.loading.LoadPriority;
import com.service.server.ITransactionResponse;
import com.service.server.incoming.alliance.AllianceBaselineResponse;
import com.service.server.incoming.alliance.AllianceGenericResponse;
import com.service.server.incoming.alliance.AllianceInviteResponse;
import com.service.server.incoming.alliance.AllianceRosterResponse;
import com.service.server.incoming.alliance.PublicAlliancesResponse;
import com.service.server.incoming.battle.BattleDataResponse;
import com.service.server.incoming.battle.BattleDebugLinesResponse;
import com.service.server.incoming.battle.BattleHasEndedResponse;
import com.service.server.incoming.battle.BattleParticipantInfo;
import com.service.server.incoming.battlelog.BattleLogDetailsResponse;
import com.service.server.incoming.battlelog.BattleLogListResponse;
import com.service.server.incoming.chat.ChatBaselineResponse;
import com.service.server.incoming.data.ActiveDefenseData;
import com.service.server.incoming.data.ActiveDefenseHitData;
import com.service.server.incoming.data.AreaAttackData;
import com.service.server.incoming.data.AreaAttackHitData;
import com.service.server.incoming.data.BattleData;
import com.service.server.incoming.data.BattleDebuff;
import com.service.server.incoming.data.BattleEntityData;
import com.service.server.incoming.data.BeamAttackData;
import com.service.server.incoming.data.DebuffMapByWeapon;
import
```

com.service.server.incoming.data.DroneAttackData;
import com.service.server.incoming.data.ProjectileAttackData;
import com.service.server.incoming.data.RemovedAttackData;
import com.service.server.incoming.data.RemovedObjectData;
import com.service.server.incoming.data.SectorBattleData;
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorEntityUpdateData;
import com.service.server.incoming.data.SectorObjectiveData;
import com.service.server.incoming.data.SectorOrderData;
import com.service.server.incoming.data.WeaponData;
import com.service.server.incoming.leaderboard.LeaderboardResponse;
import com.service.server.incoming.leaderboard.PlayerProfileResponse;
import com.service.server.incoming.leaderboard.WarfrontUpdateResponse;
import com.service.server.incoming.mail.MailDetailResponse;
import com.service.server.incoming.mail.MailInboxResponse;
import com.service.server.incoming.mail.MailUnreadResponse;
import com.service.server.incoming.proxy.ProxyBattleDisconnectedResponse;
import com.service.server.incoming.proxy.ProxySectorDisconnectedResponse;
import com.service.server.incoming.proxy.ProxyStarbaseDisconnectedResponse;
import com.service.server.incoming.sector.SectorAlwaysVisibleBaselineResponse;
import com.service.server.incoming.sector.SectorAlwaysVisibleUpdateResponse;
import com.service.server.incoming.sector.SectorBaselineResponse;
import com.service.server.incoming.sector.SectorFleetTravelAlertResponse;
import com.service.server.incoming.sector.SectorUpdateResponse;
import com.service.server.incoming.starbase.StarbaseAchievementsResponse;
import com.service.server.incoming.starbase.StarbaseAllScoresResponse;
import com.service.server.incoming.starbase.StarbaseAvailableRerollResponse;
import com.service.server.incoming.starbase.StarbaseBaselineResponse;
import com.service.server.incoming.starbase.StarbaseBattleAlertResponse;
import com.service.server.incoming.starbase.StarbaseBountyRewardResponse;
import com.service.server.incoming.starbase.StarbaseDailyResponse;
import com.service.server.incoming.starbase.StarbaseDailyRewardResponse;
import com.service.server.incoming.starbase.StarbaseFleetDockedResponse;
import com.service.server.incoming.starbase.StarbaseGetPaywallPayoutsResponse;
import com.service.server.incoming.starbase.StarbaseInstancedMissionAlertResponse;
import com.service.server.incoming.starbase.StarbaseMissionCompleteResponse;
import com.service.server.incoming.starbase.StarbaseMotdListResponse;
import com.service.server.incoming.starbase.StarbaseMoveStarbaseResponse;
import com.service.server.incoming.starbase.StarbaseOfferRedeemedResponse;
import com.service.server.incoming.starbase.StarbaseRerollChanceResultResponse;
import com.service.server.incoming.starbase.StarbaseRerollReceivedResultResponse;
import com.service.server.incoming.starbase.StarbaseUnavailableRerollResponse;
import com.service.server.incoming.universe.UniverseNeedCharacterCreateResponse;
import com.service.server.incoming.universe.UniverseSectorListResponse;
import com.service.server.outgoing.alliance.AllianceRequestBaselineRequest;
import com.service.server.outgoing.alliance.AllianceSendInviteRequest;
import com.service.server.outgoing.battle.BattleAttackOrderRequest;
import com.service.server.outgoing.battle.BattleMoveOrderRequest;
import com.service.server.outgoing.battle.BattleToggleModuleOrderRequest;
import com.service.server.outgoing.battlelog.BattleLogDetailRequest;
import

```
com.service.server.outgoing.battlelog.BattleLogListRequest;
import com.service.server.outgoing.chat.ChatReportChatRequest;
import com.service.server.outgoing.leaderboard.LeaderboardRequest;
import com.service.server.outgoing.leaderboard.LeaderboardRequestPlayerProfileRequest;
import com.service.server.outgoing.mail.MailDeleteMailRequest;
import com.service.server.outgoing.mail.MailReadMailRequest;
import com.service.server.outgoing.mail.MailRequestInboxRequest;
import com.service.server.outgoing.mail.MailSendAllianceMailRequest;
import com.service.server.outgoing.mail.MailSendMailRequest;
import com.service.server.outgoing.proxy.ProxyConnectToSectorRequest;
import com.service.server.outgoing.proxy.ProxyReportCrashRequest;
import com.service.server.outgoing.sector.SectorOrderRequest;
import com.service.server.outgoing.sector.SectorRequestBaselineRequest;
import com.service.server.outgoing.starbase.StarbaseAllScoresRequest;
import com.service.server.outgoing.starbase.StarbaseBookmarkDeleteRequest;
import com.service.server.outgoing.starbase.StarbaseBookmarkSaveRequest;
import com.service.server.outgoing.starbase.StarbaseBookmarkUpdateRequest;
import com.service.server.outgoing.starbase.StarbaseClaimAchievementRewardRequest;
import com.service.server.outgoing.starbase.StarbaseMintNFTRequest;
import com.service.server.outgoing.starbase.StarbaseClaimDailyRequest;
import com.service.server.outgoing.starbase.StarbaseGetPaywallPayoutsRequest;
import com.service.server.outgoing.starbase.StarbaseMotDReadRequest;
import com.service.server.outgoing.starbase.StarbaseRequestAchievementsRequest;
import com.service.server.outgoing.starbase.StarbaseVerifyPaymentRequest;
import com.ui.modal.server.ClientCrashView;
import com.ui.modal.server.DisconnectedView;
import com.util.AllegianceUtil;
import com.util.BattleUtils;
import com.util.TimeLog;

import flash.events.IEventDispatcher;
import flash.geom.Point;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.tick.ITickProvider;
import org.parade.core.IViewFactory;
import org.parade.core.IViewStack;
import org.parade.core.ViewEvent;
import org.shared.ObjectPool;

import flash.utils.Dictionary;

/**
*
* @author Phillip Reagan
*/
public class GameController
{
```

```
private var _assetModel:AssetModel;
private var _attackFactory:IAttackFactory;
private var _battleModel:BattleModel;
private var _eventDispatcher:IEventDispatcher;
private var _fleetModel:FleetModel;
private var _fteController:FTEController;
private var _inFTE:Boolean = false;
private var _mailModel:MailModel;
private var _missionModel:MissionModel;
private var _blueprintModel:BlueprintModel;
private var _playerModel:PlayerModel;
private var _battleLogModel:BattleLogModel;
private var _presenter:IGamePresenter;
private var _prototypeModel:PrototypeModel;
private var _sectorFactory:ISectorFactory;
private var _sectorModel:SectorModel;
private var _shipFactory:IShipFactory;
private var _soundController:SoundController;
private var _starbaseFactory:IStarbaseFactory;
private var _starbaseModel:StarbaseModel;
private var _viewFactory:IViewFactory;
private var _viewStack:IViewStack;
private var _vfxFactory:IVFXFactory;
private var _warfrontModel:WarfrontModel;
private var _leaderboardModel:LeaderboardModel;
private var _allianceModel:AllianceModel;
private var _motdModel:MotDModel;
private var _motdDailyModel:MotDDailyRewardModel;
private var _achievementModel:AchievementModel;

private var _firstTimeInit:Boolean = true;
private var _game:Game;
private var _serverController:ServerController;
private var _tickProvider:ITickProvider;
private var _transactionController:TransactionController;
private var _settingsController:SettingsController;
private var _chatController:ChatController;
private var _eventController:EventController;

private var _baseRelocatedTitle:String = 'CodeString.Toast.BaseRelocated.Title';
private var _baseRelocatedBody:String = 'CodeString.Toast.BaseRelocated.Body';

private var _alreadyRecieved:String = 'CodeString.Toast.AlreadyReceived';
private var _hasDocked:String = 'CodeString.Toast.HasDocked';

protected const _logger:ILogger = getLogger('GameController');

/**
*
*
```

```
@param game
* @param tickProvider
*/
public function GameController( game:Game, tickProvider:ITickProvider )
{
_game = game;
_tickProvider = tickProvider;
Application.onError.add(globalErrorHandler);
}

/**
*
* @param time The amount of time that has passed since the start of the last game loop
*/
public function onTick( time:Number ):void
{
if (Application.STATE == StateEvent.GAME_STARBASE)
_game.update(.033);
else
_game.update(_serverController.updateSimulationTime(time));
_viewStack.update(time);
}


//==========================================================================================
//******************************************************************************************
// BATTLE
//******************************************************************************************

//==========================================================================================

public function handleBattleDataResponse( delta:BattleDataResponse ):void
{
var response:BattleData = BattleData.globalInstance;
if (!response.hasBeenBaselined && !delta.isBaseline)
{
// we need a baseline, but this is an update. Ignore it
return;
}

response.decodeResponse(delta);
if (delta.isBaseline)
{
_logger.debug(' -- Received BattleDataResponse - baseline');

if (Application.STATE == StateEvent.GAME_BATTLE_INIT)
{
//set the faction of the battle
_sectorModel.updateSector(response.sector);

_battleModel.alloy
```

```
= response.alloy;
_battleModel.baseOwnerID = response.baseOwner;
_battleModel.battleStartTick = response.battleStartTick;
_battleModel.credits = response.credits;
_battleModel.energy = response.energy;
_battleModel.synthetic = response.synthetic;
_battleModel.isBaseCombat = response.isBaseCombat;
_battleModel.isInstancedMission = response.isInstancedMission;
_battleModel.missionID = response.missionPersistence;
_battleModel.mapSizeX = response.maxSizeX;
_battleModel.mapSizeY = response.maxSizeY;

_battleModel.galacticName = response.galacticName;
_battleModel.backgroundId = response.backgroundId;
_battleModel.planetId = response.planetId;
_battleModel.moonQuantity = response.moonQuantity;
_battleModel.asteroidQuantity = response.asteroidQuantity;
_battleModel.appearanceSeed = response.appearanceSeed;

for (participantIndex = 0; participantIndex < response.participants.length; ++participantIndex)
{
var id:String = response.participants[participantIndex].id;
var level:int = response.participants[participantIndex].level;
_battleModel.participantRatings[id] = level;
_battleModel.addParticipant(id);
}

if (response.players.length > 0)
battleAddPlayers(response.players);
var player:PlayerVO = _playerModel.getPlayer(CurrentUser.id);
if (player)
{
if(player.faction != "")
CurrentUser.battleFaction = player.faction;
}
if (response.entities.length > 0)
battleShowEntities(response.entities);
if (response.deadEntities.length > 0)
battleShowEntities(response.deadEntities);
if (response.areaAttacks.length > 0)
fireAreaAttacks(response.areaAttacks);
if (response.projectileAttacks.length > 0)
fireProjectiles(response.projectileAttacks);
if (response.beamAttacks.length > 0)
fireBeams(response.beamAttacks);

if (response.isBaseCombat)
{
_battleModel.baseFactionColor =
AllegianceUtil.instance.getFactionColor(_playerModel.getPlayer(_battleModel.baseOwnerID).faction);
//create
```

```
the starbase
_starbaseFactory.createStarbasePlatform(_battleModel.baseOwnerID);

StarbaseSystem(_game.getSystem(StarbaseSystem)).depthSort(StarbaseSystem.DEPTH_SORT_ALL);
}

//send out the battle state event
_eventDispatcher.dispatchEvent(new StateEvent(StateEvent.GAME_BATTLE));
handleBattleState(response);
}

TimeLog.endTimeLog(TimeLogEnum.SERVER_GAME_DATA, "battle");
}

// update
if (Application.STATE != StateEvent.GAME_BATTLE)
{
var msg:ProxyReportCrashRequest =
ProxyReportCrashRequest(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_REPORT_CRASH));
msg.dataStr = 'Received BattleUpdate before baseline';
_serverController.send(msg);
return;
}

_battleModel.battleEndTick = response.battleEndTick;
//create new
if (response.players.added.length > 0)
battleAddPlayers(response.players.added);
if (response.entities.added.length > 0)
battleShowEntities(response.entities.added);
if (response.droneAttacks.added.length > 0)
fireDrones(response.droneAttacks.added);
if (response.areaAttacks.added.length > 0)
fireAreaAttacks(response.areaAttacks.added);
if (response.projectileAttacks.added.length > 0)
fireProjectiles(response.projectileAttacks.added);
if (response.beamAttacks.added.length > 0)
fireBeams(response.beamAttacks.added);

//update



var health:Health;
var i:int;
var update:BattleEntityData;
var shield:Shield;
var attack:Attack;
var vcList:VCList;
for
```

```
(i = 0; i < response.entities.modified.length; i++)
{
update = response.entities.modified[i];
entity = _game.getEntity(update.id);
if (entity)
{
//Deal with the added, modded and removed maps inside the map
for each (var modByWep:DebuffMapByWeapon in update.debuffs.modified)
{
var tray:DebuffTray;
if (entity.has(DebuffTray))
tray = entity.get(DebuffTray);
else
{
tray = ObjectPool.get(DebuffTray);
tray.init();
}
for (var addDebuffKey:String in modByWep.added)
{
var addDebuff:BattleDebuff = modByWep.added[addDebuffKey];
var assetVO:AssetVO = _assetModel.getEntityData(addDebuff.prototype);
if (!entity.has(DebuffTray))
{
entity.add(tray);
tray.addDebuff(addDebuffKey, assetVO, addDebuff.stackCount);
vcList = entity.get(VCList);
vcList.addComponentType(TypeEnum.DEBUFF_TRAY);
} else
tray.addDebuff(addDebuffKey, assetVO, addDebuff.stackCount);
}

for each (var modDebuff:BattleDebuff in modByWep.modified)
{
tray.addDebuff(null, _assetModel.getEntityData(modDebuff.prototype), modDebuff.stackCount);
}

for each (var remDebuff:RemovedObjectData in modByWep.removed)
{
//ensure the tray is still on the ship by grabbing it every time
tray = entity.get(DebuffTray);
if (tray)
{
tray.removeDebuff(remDebuff.id);

if (tray.isDebuffsEmpty())
{
vcList = entity.get(VCList);
vcList.removeComponentType(TypeEnum.DEBUFF_TRAY);
entity.remove(DebuffTray);
ObjectPool.give(tray);
}
```

```
}
}
}

health = Health(entity.get(Health));
if (health && health.currentHealth != update.currentHealth)
{
health.currentHealth = update.currentHealth;
}

shield = entity.get(Shield);
if (shield)
{
shield.enabled = update.shieldsEnabled;
shield.currentStrength = update.shieldsCurrentHealth;
}

if (update.selectedTargetId != "UNSET")
{
attack = entity.get(Attack);
if (attack)
attack.targetID = update.selectedTargetId;
} else if (update.organicTargetId != "UNSET")
{
attack = entity.get(Attack);
if (attack)
attack.targetID = update.organicTargetId;
}

var modules:Modules = entity.get(Modules);
if (modules)
{
for (var weaponidx:int = 0; weaponidx < update.weapons.modified.length; ++weaponidx)
{
var weapon:WeaponData = WeaponData(update.weapons.modified[weaponidx]);
modules.moduleStates[weapon.moduleIdx] = weapon.weaponState;
}
}
}
}

//Handle Area Attack Collisions
var area:AreaAttackHitData;
var aEntity:Entity;
if (response.areaAttackHits.length > 0 && Application.STARLING_ENABLED)
{
var len:int = response.areaAttackHits.length;
//limiting this to under 7 hits for performance concerns.
if (len < 7)
{
```

```
for (i = 0; i < len; i++)
{
area = response.areaAttackHits[i];
aEntity = _game.getEntity(area.attackId);

_vfxFactory.createHit(aEntity, aEntity, area.locationX, area.locationY, false, false);
}
}
}

//remove projectiles
var entity:Entity;
var position:Position;
var removedEntity:RemovedObjectData;
var removedAttack:RemovedAttackData;
var removedAttacks:Array =
response.areaAttacks.removed.concat(response.beamAttacks.removed,
response.droneAttacks.removed, response.projectileAttacks.removed);
for (i = 0; i < removedAttacks.length; i++)
{
removedAttack = removedAttacks[i];
entity = _game.getEntity("Attack" + String(removedAttack.id));
if (entity)
{
switch (removedAttack.reason)
{
case RemoveReasonEnum.AttackComplete:
_soundController.playSound(AudioEnum.AFX_WEAPON_HIT, 0.5);
if (_vfxFactory.createHit(entity, entity, removedAttack.x, removedAttack.y, false, true) == null)
{
_attackFactory.destroyAttack(entity);
}
break;
case RemoveReasonEnum.Intercepted:
var activeDefenseHit:ActiveDefenseHitData = response.adHits[entity.id];
if (activeDefenseHit)
{
aEntity = _game.getEntity(activeDefenseHit.owningShip);
if (aEntity)
_attackFactory.createActiveDefenseInterceptor(aEntity, activeDefenseHit.attachPoint,
removedAttack.x, removedAttack.y);
}
_attackFactory.destroyAttack(entity);
break;
case RemoveReasonEnum.ShieldComplete:
_soundController.playSound(AudioEnum.AFX_BARRIER_HIT, 0.5);
if (_vfxFactory.createHit(entity, entity, removedAttack.x, removedAttack.y, true, true) == null)
{
_attackFactory.destroyAttack(entity);
}
```

```
break;
default:
_attackFactory.destroyAttack(entity);
break;
}
}
}

//remove entities
for (i = 0; i < response.entities.removed.length; i++)
{
removedEntity = response.entities.removed[i];
destroyEntity(removedEntity.id, removedEntity.reason);
}

for (var participantIndex:int = 0; participantIndex < response.participants.added.length;
++participantIndex)
{
var participant:BattleParticipantInfo =
BattleParticipantInfo(response.participants.added[participantIndex]);
_battleModel.participantRatings[participant.id] = participant.level;
_battleModel.addParticipant(participant.id);
_battleModel.reconnect();
}

//remove players
if (response.players.removed.length)
battleRemovePlayers(response.players.removed);

handleBattlePositionUpdateResponse(response);

if (response.battleStateChanged)
{
handleBattleState(response);
}
}


/**
*
* @param response
*/
public function handleBattleDebugLinesResponse( response:BattleDebugLinesResponse ):void
{
var dlSystem:DebugLineSystem = DebugLineSystem(_game.getSystem(DebugLineSystem));
if (!dlSystem)
return;

dlSystem.addLine(response.debugLines);
dlSystem.removeLine(response.removedLines);
```

```
}

/**
*
* @param response
*/
public function handleBattlePositionUpdateResponse( response:BattleData ):void
{
if (Application.STATE != StateEvent.GAME_BATTLE)
{
//var msg:ProxyReportCrashRequest =
ProxyReportCrashRequest(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_REPORT_CRASH));
//msg.dataStr = 'Received BattlePositionUpdate before baseline';
//_serverController.send(msg);
return;
}

var animation:Animation;
var i:int = 0;
var detail:Detail;
var entity:Entity;
var move:Move;
var rotation:Number;
var turret:Entity;
var vcList:VCList;
for (i = 0; i < response.entities.modified.length; i++)
{
var battleEntity:BattleEntityData = response.entities.modified[i];
entity = _game.getEntity(battleEntity.id);
if (entity)
{
detail = entity.get(Detail);
move = entity.get(Move);
if (move)
{
move.addUpdate(battleEntity.location.x, battleEntity.location.y, battleEntity.velocity.x,
battleEntity.velocity.y,
battleEntity.rotation,
response.tick,
response.tick + 1);
}

//update turret rotations
if (detail.type == TypeEnum.POINT_DEFENSE_PLATFORM &&
battleEntity.weapons.modified.length > 0)
{
vcList = entity.get(VCList);
turret = vcList.getComponent(TypeEnum.STARBASE_TURRET);
if
```

```
(turret)
{
// Determine sprite frame to use based on angle
rotation = battleEntity.weapons.modified[0].rotation;
var num:int = rotation * (44.0 / 256.0) | 0;
animation = turret.get(Animation);
animation.frame = num;
if (animation.render && animation.spritePack)
animation.render.updateFrame(animation.spritePack.getFrame(animation.label, num),
animation);
}
}
}
}

for (i = 0; i < response.droneAttacks.modified.length; i++)
{
var droneAttack:DroneAttackData = response.droneAttacks.modified[i];
entity = _game.getEntity(droneAttack.attackId);
if (entity)
{
Move(entity.get(Move)).addUpdate(droneAttack.location.x, droneAttack.location.y, 0, 0,
droneAttack.rotation, response.tick, response.tick + 1);
var drone:Drone = entity.get(Drone);
drone.isOribiting = droneAttack.isOrbiting;
drone.targetID = droneAttack.targetEntityId;
}
}

for (i = 0; i < response.beamAttacks.modified.length; i++)
{
var beamAttack:BeamAttackData = response.beamAttacks.modified[i];
entity = _game.getEntity(beamAttack.attackId);
if (entity)
{
var beam:Beam = entity.get(Beam);
beam.targetID = beamAttack.targetEntityId;
beam.targetAttachPoint = beamAttack.targetAttachPoint;
beam.targetScatterX = beamAttack.targetScatterX;
beam.targetScatterY = beamAttack.targetScatterY;
beam.attackHit = beamAttack.attackHit;
beam.hitLocationX = beamAttack.hitLocation.x;
beam.hitLocationY = beamAttack.hitLocation.y;
beam.hitTarget = beamAttack.hitTarget;

//if the attack hit the target set shownHit = false so that the hit animation will be shown next loop
if (beam.attackHit)
beam.visibleHitCounter--;
}
}


for
```

```
(i = 0; i < response.projectileAttacks.modified.length; i++)
{
var projectileAttack:ProjectileAttackData = response.projectileAttacks.modified[i];
entity = _game.getEntity(projectileAttack.attackId);
if (entity)
{
// Iso crunch the rotation of guided attacks
var rot:Number = projectileAttack.rotation;
rot = BattleUtils.instance.isoCrunchAngle(rot);

// TODO - projectile velocity is constant and known... dig it up and plug it in here to allow
projectiles to sim beyond updates if there's lag
Move(entity.get(Move)).addUpdate(projectileAttack.location.x, projectileAttack.location.y, 0, 0,
rot, response.tick, response.tick + 1);
}
}
}

public function handleBattleEnded( response:BattleHasEndedResponse ):void
{
_battleModel.finished = true;
_battleModel.wonLastBattle = (CurrentUser.id in response.victors);
_eventDispatcher.dispatchEvent(new BattleEvent(BattleEvent.BATTLE_ENDED, null,
response));
if (_inFTE)
{
_inFTE = false;
_fteController.nextStep();
}
}

/**
*
* @param response
*/
public function handleBattleState( response:BattleData ):void
{
if (response.battleState == 3 || response.battleState == 4)
{
_eventDispatcher.dispatchEvent(new BattleEvent(BattleEvent.BATTLE_COUNTDOWN, null,
response));
}
if (response.battleState == 5)
{
_battleModel.battleEndTick = response.battleEndTick;
_battleModel.battleStartTick = response.battleStartTick;
_eventDispatcher.dispatchEvent(new BattleEvent(BattleEvent.BATTLE_STARTED, null,
response));
}
}

/**
```

```
*
* @param id
* @param targetID
*/
public function battleAttackShip( id:String, targetID:String, moveToTarget:Boolean = false ):void
{
var order:BattleAttackOrderRequest =
BattleAttackOrderRequest(_serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_ATTACK_ORDER));
order.entityID = id;
order.targetID = targetID;
order.issuedTick = ServerController.SIMULATED_TICK;
order.subSystemTarget = -1; // TODO - add subsystem targeting controls
order.moveToTarget = moveToTarget;
_serverController.send(order);
}

/**
*
* @param id
* @param x
* @param y
* @param startTick
*/
public function battleMoveShip( id:String, x:int, y:int, startTick:int ):void
{
var order:BattleMoveOrderRequest =
BattleMoveOrderRequest(_serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_MOVE_ORDER));
order.entityID = id;
order.targetX = x;
order.targetY = y;
order.startTick = startTick;
_serverController.send(order);
}

private function battleShowEntities( entities:Array ):void
{
var entityData:BattleEntityData;
var entity:Entity;
var forcefield:Vector.<BattleEntityData>;
var position:Point = new Point();
for (var i:int = 0; i < entities.length; i++)
{
entityData = BattleEntityData(entities[i]);
_starbaseFactory.setBaseFaction(entityData.factionId);
if (!_game.getEntity(entityData.id))
{
switch (entityData.type)
{
```

```
case BattleEntityTypeEnum.SHIP:
_battleModel.addBattleEntity(entityData);
if (entityData.currentHealth > 0)
entity = _shipFactory.createShip(entityData);
else if(entityData.currentHealth == 0)
{
entity = _shipFactory.createShip(entityData);
_shipFactory.destroyShip(entity);
}
break;
case BattleEntityTypeEnum.PYLON:
case BattleEntityTypeEnum.BUILDING:
_battleModel.addBattleEntity(entityData);
entity = _starbaseFactory.createBattleBuilding(entityData);
break;
case BattleEntityTypeEnum.PLATFORM:
entity = _starbaseFactory.createBattleBaseItem(entityData);
break;
case BattleEntityTypeEnum.FORCEFIELD:
if (entityData.currentHealth > 0)
{
if (forcefield == null)
forcefield = new Vector.<BattleEntityData>;
forcefield.push(entityData);
}
break;
default:
trace("making", entityData.type);
break;
}
}
}

if (forcefield)
{
var player:PlayerVO = _playerModel.getPlayer(forcefield[0].ownerId);
if (player)
{
var color:uint = AllegianceUtil.instance.getFactionColor(player.faction);
var pylonA:Pylon;
var pylonB:Pylon;
for (i = 0; i < forcefield.length; i++)
{

pylonA = Pylon(_game.getEntity(forcefield[i].connectedPylons[0]).get(Pylon));
pylonB = Pylon(_game.getEntity(forcefield[i].connectedPylons[1]).get(Pylon));
_starbaseFactory.createForcefield(forcefield[i].id, pylonA, pylonB, color);
}
}
}
```

```
}


//=============================================================================
//*****************************************************************************
// PROXY
//*****************************************************************************

//=============================================================================

public function handleProxyBattleDisconnected( response:ProxyBattleDisconnectedResponse
):void
{
if (!_battleModel.finished)
{
if (Application.STATE == StateEvent.GAME_BATTLE ||
Application.STATE == StateEvent.GAME_BATTLE_INIT)
{
if (_battleModel.oldGameState == StateEvent.GAME_STARBASE)
{
var starbaseEvent:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
_eventDispatcher.dispatchEvent(starbaseEvent);
} else
{
var event:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, null, null,
_battleModel.focusLocation.x, _battleModel.focusLocation.y);
_eventDispatcher.dispatchEvent(event);
}
_chatController.addSystemMessage("Your Battle was terminated.\n");
}
}
}

public function handleProxySectorDisconnected( response:ProxySectorDisconnectedResponse
):void
{
if (Application.STATE == StateEvent.GAME_SECTOR ||
Application.STATE == StateEvent.GAME_SECTOR_INIT)
{
var event:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE, null);
_eventDispatcher.dispatchEvent(event);
_chatController.addSystemMessage("The Sector is restarting.\n");

if (_inFTE)
showDisconnect('CONNECTION LOST', 'Your connection was no match for the
Imperium!\nPlease refresh your browser.\n\nError Message: 3027 Sector Error');
}
}

public
```

```
function handleProxyStarbaseDisconnected( response:ProxyStarbaseDisconnectedResponse
):void
{
if (Application.STATE == StateEvent.GAME_STARBASE && !_inFTE)
{
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR,
_sectorModel.sectorID, null);
_eventDispatcher.dispatchEvent(sectorEvent);
} else if (_inFTE)
showDisconnect('CONNECTION LOST', 'Your connection was no match for the
Imperium!\nPlease refresh your browser.\n\nError Message: 3026 Starbase Error');
}

private function showDisconnect( title:String = 'CONNECTION LOST', message:String = 'Your
connection was no match for the Imperium!\n\nPlease refresh your browser.' ):void
{
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
var nDisconnectView:DisconnectedView =
DisconnectedView(_viewFactory.createView(DisconnectedView));
nDisconnectView.titleText = title;
nDisconnectView.messageText = message;
viewEvent.targetView = nDisconnectView;
_eventDispatcher.dispatchEvent(viewEvent);
}



//==========================================================================
//**************************************************************************************************
// UNIVERSE
//**************************************************************************************************

//==========================================================================


public function handleUniverseNeedCharacterCreateResponse(
response:UniverseNeedCharacterCreateResponse ):void
{
var serverEvent:ServerEvent = new ServerEvent(ServerEvent.NEED_CHARACTER_CREATE);
_eventDispatcher.dispatchEvent(serverEvent);
}


//==========================================================================
//**************************************************************************************************
// SECTOR
//**************************************************************************************************

//==========================================================================

/**
```

```
*
* @param response
*/
public function handleSectorBaselineResponse( response:SectorBaselineResponse ):void
{
_logger.debug(' -- Received SectorBaselineResponse');

if (response.entities.length > 0)
sectorShowEntities(response.entities, false);
if (response.orders.length > 0)
sectorAssignOrders(response.orders, false);
if (response.battles.length > 0)
sectorAssignBattles(response.battles);

//send out the sector state event
_eventDispatcher.dispatchEvent(new StateEvent(StateEvent.GAME_SECTOR));
TimeLog.endTimeLog(TimeLogEnum.SERVER_GAME_DATA, "sector");
}

/**
*
* @param response
*/
public function handleSectorAlwaysVisibleBaselineResponse(
response:SectorAlwaysVisibleBaselineResponse ):void
{
_sectorModel.updateSector(response.sector);
if (response.players.length > 0)
addPlayers(response.players);
if (response.entities.length > 0)
sectorShowEntities(response.entities, true);
if (response.orders.length > 0)
sectorAssignOrders(response.orders, true);
if (response.battles.length > 0)
sectorAssignBattles(response.battles);
//if (response.objectives.length > 0)
//sectorShowObjectives(response.objectives);
}

/**
*
* @param response
*/
public function handleSectorUpdateResponse( response:SectorUpdateResponse ):void
{
if (response.entities.length > 0)
sectorShowEntities(response.entities, false);
if (response.orders.length > 0)
sectorAssignOrders(response.orders, false);
if
```

```
(response.battles.length > 0)
sectorAssignBattles(response.battles);
if (response.entityUpdates.length > 0)
sectorUpdateEntities(response.entityUpdates);

//remove entities
var removed:RemovedObjectData;
for (var i:int = 0; i < response.removedEntities.length; i++)
{
removed = response.removedEntities[i];
var entity:Entity = _game.getEntity(removed.id);
if (entity)
{
// Don't destroy anything owned by the player, since the AlwaysVisible update is authoritative
over those.
if (entity.has(Owned))
continue;
destroyEntity(removed.id, response.removedEntities[i].reason);
}
}

//remove battles
if (response.removedBattles.length > 0)
sectorRemoveBattles(response.removedBattles);
}

/**
*
* @param response
*/
public function handleSectorAlwaysVisibleUpdateResponse(
response:SectorAlwaysVisibleUpdateResponse ):void
{
if (response.players.length > 0)
addPlayers(response.players);
if (response.entities.length > 0)
sectorShowEntities(response.entities, true);
if (response.orders.length > 0)
sectorAssignOrders(response.orders, true);
if (response.battles.length > 0)
sectorAssignBattles(response.battles);
if (response.entityUpdates.length > 0)
sectorUpdateEntities(response.entityUpdates);
if (response.objectives.length > 0)
sectorShowObjectives(response.objectives);

//remove entities
var removed:RemovedObjectData;
for (var i:int = 0; i < response.removedEntities.length; i++)
{
removed
```

```
= response.removedEntities[i];
destroyEntity(removed.id, response.removedEntities[i].reason);
}

//remove players
if (response.removedPlayers.length > 0)
removePlayers(response.removedPlayers);
//remove battles
if (response.removedBattles.length > 0)
sectorRemoveBattles(response.removedBattles);

for (i = 0; i < response.removedObjectives.length; i++)
{
removed = response.removedObjectives[i];
destroyEntity(removed.id, response.removedObjectives[i].reason);
}

}

private function sectorAssignBattles( battles:Vector.<SectorBattleData> ):void
{
var entity:Entity;
var battle:SectorBattleData;
var fleetVO:FleetVO;
for (var i:int = 0; i < battles.length; i++)
{
battle = battles[i];
if (!_game.getEntity(battle.id))
{
for (var j:int = 0; j < battle.participantFleets.length; j++)
{
entity = _game.getEntity(battle.participantFleets[j]);
if (entity)
{
Attack(entity.get(Attack)).battleServerAddress = battle.serverIdentifier;
Attack(entity.get(Attack)).inBattle = true;
Attack(entity.get(Attack)).battle = battle;
}
}
if (battle.participantBase && battle.participantBase != '')
{
entity = _game.getEntity(battle.participantBase);
if (entity)
{
Attack(entity.get(Attack)).battleServerAddress = battle.serverIdentifier;
Attack(entity.get(Attack)).inBattle = true;
Attack(entity.get(Attack)).battle = battle;
}
}

_vfxFactory.createAttackIcon(battle);
```

```
}
}
}

private function sectorRemoveBattles( battles:Vector.<RemovedObjectData> ):void
{
var attack:Attack;
var battle:SectorBattleData;
var attackIcon:Entity;
var battleEntity:Entity;
var fleetVO:FleetVO;
for (var i:int = 0; i < battles.length; i++)
{
attackIcon = _game.getEntity(battles[i].id);
if (attackIcon)
{
attack = attackIcon.get(Attack);
battle = attack.attackData;
for (var j:int = 0; j < battle.participantFleets.length; j++)
{
battleEntity = _game.getEntity(battle.participantFleets[j]);
if (battleEntity)
{
Attack(battleEntity.get(Attack)).inBattle = false;
Attack(battleEntity.get(Attack)).battle = null;
}
}
if (battle.participantBase && battle.participantBase != '')
{
battleEntity = _game.getEntity(battle.participantBase);
if (battleEntity)
{
Attack(battleEntity.get(Attack)).inBattle = false;
Attack(battleEntity.get(Attack)).battle = null;
}
}
_vfxFactory.destroyAttack(attackIcon);
}
}
}

private function sectorAssignOrders( orders:Vector.<SectorOrderData>,
includeSelfOwned:Boolean ):void
{
var entity:Entity;
var fleet:FleetVO;
var move:Move;
var order:SectorOrderData;
var sectorInteractSystem:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
```

```
var selectedEntity:Entity = sectorInteractSystem.selected;
var shipClass:String = '';
for (var i:int = 0; i < orders.length; i++)
{
order = orders[i];
entity = _game.getEntity(order.entityId);
if (entity)
{
if (!includeSelfOwned)
{
if (entity.get(Owned))
continue;
}

fleet = _fleetModel.getFleet(entity.id);
if (fleet)
fleet.defendTarget = "";
move = entity.get(Move);
switch (order.orderType)
{
case OrderEnum.RECALL:
if (fleet)
fleet.state = FleetStateEnum.DOCKING;
Attack(entity.get(Attack)).targetID = order.targetId;
move.setPointToPoint(order.targetLocationX, order.targetLocationY, order.issuedTick,
order.finishTick);
break;
case OrderEnum.FORCED_RECALL:
if (Detail(entity.get(Detail)).ownerID == CurrentUser.id)
{
fleet = _fleetModel.getFleet(entity.id);
if (fleet)
fleet.state = FleetStateEnum.FORCED_RECALLING;
}

case OrderEnum.ATTACK:
case OrderEnum.FORCE_ATTACK:
case OrderEnum.SALVAGE:
case OrderEnum.TRANS_GATE_TRAVEL:
case OrderEnum.WAYPOINT_TRAVEL:
Attack(entity.get(Attack)).targetID = order.targetId;
move.setPointToPoint(order.targetLocationX, order.targetLocationY, order.issuedTick,
order.finishTick);
break;
case OrderEnum.DEFEND:
if (fleet)
fleet.defendTarget = order.targetId;
case OrderEnum.MOVE:
case OrderEnum.REMOTE_MOVE:
case
```

```
OrderEnum.TACKLE:
case OrderEnum.HALT:
if (Detail(entity.get(Detail)).ownerID == CurrentUser.id)
{
shipClass = Detail(entity.get(Detail)).prototypeVO ?
Detail(entity.get(Detail)).prototypeVO.itemClass : '';

//Play ship move sound based on size
switch (shipClass)
{

case TypeEnum.FIGHTER:
case TypeEnum.TRANSPORT:
SoundController.instance.playSound(AudioEnum.AFX_ENGINE_ION, 0.5, 0, 1);
break;
case TypeEnum.HEAVY_FIGHTER:
SoundController.instance.playSound(AudioEnum.AFX_ENGINE_HEAVY_ION, 0.5, 0, 1);
break;
case TypeEnum.CORVETTE:
SoundController.instance.playSound(AudioEnum.AFX_ENGINE_IMPULSE, 0.5, 0, 1);
break;
case TypeEnum.DESTROYER:
SoundController.instance.playSound(AudioEnum.AFX_ENGINE_HEAVY_IMPULSE, 0.5, 0, 1);
break;
case TypeEnum.BATTLESHIP:
SoundController.instance.playSound(AudioEnum.AFX_ENGINE_FUSION, 0.5, 0, 1);
break;
case TypeEnum.DREADNOUGHT:
SoundController.instance.playSound(AudioEnum.AFX_ENGINE_HEAVY_FUSION, 0.5, 0, 1);
break;
}
}
Attack(entity.get(Attack)).targetID = null;
move.setPointToPoint(order.targetLocationX, order.targetLocationY, order.issuedTick,
order.finishTick);
break;
}

if (fleet && fleet.state == FleetStateEnum.DOCKING && order.orderType !=
OrderEnum.RECALL)
fleet.state = FleetStateEnum.OUT;

if (selectedEntity != null && entity.get(Owned) && selectedEntity == entity)
_fleetModel.updateFleet(_fleetModel.getFleet(entity.id));
}
}
sectorInteractSystem.showSelector();
}

/**
*
```

```
Called to update misc. attributes on entities within a sector.
* @param updates A list of entities to update
*/
private function sectorUpdateEntities( updates:Vector.<SectorEntityUpdateData> ):void
{
var cargo:Cargo;
var detail:Detail;
var entity:Entity;
var entityUpdate:SectorEntityUpdateData;
var position:Position;
var vcList:VCList;
var vcSystem:VCSystem = VCSystem(_game.getSystem(VCSystem));
for (var i:int = 0; i < updates.length; i++)
{
entityUpdate = updates[i];
entity = _game.getEntity(entityUpdate.id);

if (!entity)
continue;

detail = entity.get(Detail);
if (vcSystem)
{
position = entity.get(Position);
switch (detail.type)
{
case TypeEnum.STARBASE_SECTOR_IGA:
vcList = entity.get(VCList);
if (entityUpdate.bubbled)
{
vcList.addComponentType(TypeEnum.STARBASE_SHIELD_IGA);
if (entityUpdate.currentHealthPct <
_prototypeModel.getConstantPrototypeByName("protectionLowDamageThreshold").getValue('value'))
_vfxFactory.createSectorExplosion(entity, position.x, position.y);
} else
vcList.removeComponentType(TypeEnum.STARBASE_SHIELD_IGA);
Attack(entity.get(Attack)).bubbled = entityUpdate.bubbled;
Animation(entity.get(Animation)).label = (entityUpdate.currentHealthPct < .25) ?
detail.assetVO.spriteName + detail.level + "DMG" : detail.assetVO.spriteName + detail.level;
break;
case TypeEnum.STARBASE_SECTOR_SOVEREIGNTY:
vcList = entity.get(VCList);
if (entityUpdate.bubbled)
{
vcList.addComponentType(TypeEnum.STARBASE_SHIELD_SOVEREIGNTY);
if (entityUpdate.currentHealthPct <
_prototypeModel.getConstantPrototypeByName("protectionLowDamageThreshold").getValue('value'))
_vfxFactory.createSectorExplosion(entity, position.x, position.y);
} else
vcList.removeComponentType(TypeEnum.STARBASE_SHIELD_SOVEREIGNTY);
Attack(entity.get(Attack)).bubbled
```

```
= entityUpdate.bubbled;
Animation(entity.get(Animation)).label = (entityUpdate.currentHealthPct < .25) ?
detail.assetVO.spriteName + detail.level + "DMG" : detail.assetVO.spriteName + detail.level;
break;
case TypeEnum.STARBASE_SECTOR_TYRANNAR:
vcList = entity.get(VCList);
if (entityUpdate.bubbled)
{
vcList.addComponentType(TypeEnum.STARBASE_SHIELD_TYRANNAR);
if (entityUpdate.currentHealthPct <
_prototypeModel.getConstantPrototypeByName("protectionLowDamageThreshold").getValue('value'))
_vfxFactory.createSectorExplosion(entity, position.x, position.y);
} else
vcList.removeComponentType(TypeEnum.STARBASE_SHIELD_TYRANNAR);
Attack(entity.get(Attack)).bubbled = entityUpdate.bubbled;
Animation(entity.get(Animation)).label = (entityUpdate.currentHealthPct < .25) ?
detail.assetVO.spriteName + detail.level + "DMG" : detail.assetVO.spriteName + detail.level;
break;
}
}

cargo = entity.get(Cargo);
if (cargo)
{
cargo.cargo = entityUpdate.cargo;

var fleetVO:FleetVO = _fleetModel.getFleet(entityUpdate.id);
if (fleetVO)
{
fleetVO.currentCargo = cargo.cargo;
_fleetModel.updateFleet(fleetVO);

_soundController.playSound(AudioEnum.AFX_GLOBAL_CARGO_COLLECT);
}
}
}
}

/**
*
* @param response
*/
public function sectorFleetTravelAlert( response:SectorFleetTravelAlertResponse ):void
{
var fleetVO:FleetVO = _fleetModel.getFleet(response.entityId);
if (fleetVO)
{
fleetVO.sector = response.sectorKey;
_fleetModel.updateFleet(fleetVO);
}
}
```

```
/**
*
* @param response
*/
public function handleUniverseSectorListResponse( response:UniverseSectorListResponse
):void
{
_sectorModel.addDestinations(response.sectors);
_sectorModel.addPrivateDestinations(response.privateSectors);
}

/**
*
* @param id
* @param x
* @param y
*/
public function sectorMoveFleet( id:String, x:int, y:int ):void
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = id;
order.orderType = OrderEnum.MOVE;
order.targetLocationX = x;
order.targetLocationY = y;
_serverController.send(order);
}

/**
*
*/
public function sectorRequestBaseline( conectToSector:Boolean = true ):void
{
if (conectToSector)
{
var connectRequest:ProxyConnectToSectorRequest =
ProxyConnectToSectorRequest(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_CONNECT_TO_SECTOR));
connectRequest.key = _sectorModel.targetSector;
_serverController.send(connectRequest);
}

var request:SectorRequestBaselineRequest =
SectorRequestBaselineRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_REQUEST_BASELINE));
var sis:SectorInteractSystem = SectorInteractSystem(_game.getSystem(SectorInteractSystem));
request.viewX = sis.sceneX;
request.viewY
```

```
= sis.sceneY;
_serverController.send(request);
}

private function sectorShowEntities( entities:Vector.<SectorEntityData>,
includeSelfOwned:Boolean ):void
{
var entityData:SectorEntityData;
for (var i:int = 0; i < entities.length; i++)
{
entityData = entities[i];
if (!includeSelfOwned && entityData.ownerId == CurrentUser.id)
continue;

var entity:Entity = _game.getEntity(entityData.id);

switch (entityData.type)
{
case SectorEntityTypeEnum.BASE:
if (!entity)
{
_sectorFactory.createSectorBase(entityData);
}
break;
case SectorEntityTypeEnum.FLEET:
if (!entity)
{
_shipFactory.createFleet(entityData);
}

if (entityData.ownerId == CurrentUser.id)
{
var fleetVO:FleetVO = _fleetModel.getFleet(entityData.id);
fleetVO.currentHealth = entityData.currentHealthPct;
fleetVO.sector = _sectorModel.sectorID;
if (entityData.state == SectorEntityStateEnum.DEFENDING)
fleetVO.defendTarget = "defending";
}
break;
case SectorEntityTypeEnum.TRANS_GATE:
if (!entity)
{
_sectorFactory.createTransgate(entityData);
}
break;
case SectorEntityTypeEnum.DEPOT:
if (!entity)
{
_sectorFactory.createDepot(entityData);
}
break;
```

```
case SectorEntityTypeEnum.DERELICT:
if (!entity)
{
_sectorFactory.createDerelict(entityData);
}
break;
}
}

_fleetModel.updateFleet(null);
}

private function sectorShowObjectives( objectives:Vector.<SectorObjectiveData> ):void
{
var objectiveData:SectorObjectiveData;
for (var i:int = 0; i < objectives.length; i++)
{
objectiveData = objectives[i];

var entity:Entity = _game.getEntity(objectiveData.missionKey);

if (!entity)
_sectorFactory.createObjective(objectiveData);

}
}


//==========================================================================
//**************************************************************************************************
// STARBASE
//**************************************************************************************************

//==========================================================================

/**
*
* @param response
*/
public function handleStarbaseBaselineResponse( response:StarbaseBaselineResponse ):void
{
if(!response.validData)
{
return;
}
_logger.debug(' -- Received StarbaseBaselineResponse [bases: {0}, buildings: {1}, fleets: {2},
ships: {3}, isUpdate:{4}, reason: {5}]',
[response.bases.length, response.buildings.length, response.fleets.length,
response.ships.length, response.update, response.updateReason]);

_starbaseModel.setBaseDirty();
```

```
if (!response.update)
_prototypeModel.setSplits(response.activeSplitPrototypes);

//handle starbases
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_BASE)
{
for (var i:int = 0; i < response.bases.length; i++)
{
_starbaseModel.importBaseData(response.bases[i], !response.update);
}
}
//handle buildings
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_BUILDING)
{
for (i = 0; i < response.buildings.length; i++)
{
_starbaseModel.importBuildingData(response.buildings[i]);
}
}
//handle buffs
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_BUFF)
{
for (i = 0; i < response.buffs.length; i++)
{
_starbaseModel.importBuffData(response.buffs[i]);
}
}

//handle blueprints
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_BLUEPRINT)
{
for (i = 0; i < response.blueprints.length; i++)
{
_blueprintModel.importPlayerBlueprints(response.blueprints[i], response.update);
}
}

//handle missions
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_MISSION)
{
for (i = 0; i < response.missions.length; i++)
{
_missionModel.importMissionData(response.missions[i]);
//see
```

```
if the fte needs any of these missions
if (_fteController.running)
_fteController.checkMissionRequired(_missionModel.currentMission);
}
}

//handle research
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_RESEARCH)
{
for (i = 0; i < response.research.length; i++)
{
_starbaseModel.importResearchData(response.research[i]);
}

if (!response.update)
_starbaseModel.addBeginnerResearch(_prototypeModel.getResearchPrototypes());
}

//handle traderoutes
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_TRADE_ROUTE)
{
for (i = 0; i < response.tradeRoutes.length; ++i)
{
_starbaseModel.importTradeRouteData(response.tradeRoutes[i]);
}
}

//handle fleets
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_FLEET)
{
for (i = 0; i < response.fleets.length; ++i)
{
_fleetModel.importFleetData(response.fleets[i]);
}
}
//handle ships
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_SHIP)
{
for (i = 0; i < response.ships.length; ++i)
{
_fleetModel.importShipData(response.ships[i]);
}
}
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_FLEET ||
response.baselineType & StarbaseBaselineResponse.BASELINE_SHIP)
_fleetModel.updateFleet(null);
```

```
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_PLAYER)
CurrentUser.addBookmarks(response.bookmarks.bookmarks);

if (!response.update)
{
_eventController.addEvents(response.activeEvents, response.upcomingEvents,
response.nowMillis);

if (Application.STATE == StateEvent.GAME_STARBASE)
IStarbasePresenter(_presenter).showBuildings();

if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_SETTING)
_settingsController.setSettings(response.settings);
}
if(response.baselineType == StarbaseBaselineResponse.BASELINE_ALL ||
response.baselineType & StarbaseBaselineResponse.BASELINE_FLEET ||
response.baselineType & StarbaseBaselineResponse.BASELINE_SHIP)
{
_fleetModel.dirty = false;
_fleetModel.maxAvailableShipSlots =
_prototypeModel.getConstantPrototypeValueByName('MaxAvailabeShipSlotsBase') +
CurrentUser.purchasedShipSlots;
}
_starbaseModel.currentBase.updateResources();
_transactionController.dataImported();
}

/**
*
* @param response
*/
public function handleStarbaseTransactionResponse( response:ITransactionResponse ):void
{
_transactionController.handleResponse(response);
}

/**
*
* @param response
*/
public function starbaseInstancedMissionAlert(
response:StarbaseInstancedMissionAlertResponse ):void
{
//see if any fleets are in battle
var fleets:Vector.<FleetVO> = _fleetModel.fleets;
var fleetVO:FleetVO;
var
```

```
inBattle:Boolean = Application.STATE == null || ((Application.STATE ==
StateEvent.GAME_BATTLE_INIT || Application.STATE == StateEvent.GAME_BATTLE) &&
!_battleModel.finished);
var newInstancedMissionCombat:Boolean = false;
var soundAlarm:Boolean = false;
var notifyFleetID:String;

//set the battle server address of center space battles
/*if (_starbaseModel.centerSpaceBase)
{
if (response.centerSpaceBaseBattle != null && response.centerSpaceBaseBattle != "")
{
if (_starbaseModel.centerSpaceBase.battleServerAddress == null)
soundAlarm = true;
_starbaseModel.centerSpaceBase.battleServerAddress = response.centerSpaceBaseBattle;
} else
_starbaseModel.centerSpaceBase.battleServerAddress = null;
}*/
//set the battle server address of homebase battles
if (response.instancedMissionBattle != null && response.instancedMissionBattle != "")
{
if (_starbaseModel.homeBase.instancedMissionAddress == null)
{
newInstancedMissionCombat = true;
soundAlarm = true;
}
_starbaseModel.homeBase.instancedMissionAddress = response.instancedMissionBattle;
} else
_starbaseModel.homeBase.instancedMissionAddress = null;

if (!_fteController.running && Application.STATE != null && !inBattle)
{
var event:StarbaseEvent;
if (newInstancedMissionCombat && response.instancedMissionBattle &&
response.instancedMissionBattle != "")
{
//notify of a home base battle
event = new StarbaseEvent(StarbaseEvent.ALERT_INSTANCED_MISSION_BATTLE,
_starbaseModel.homeBase.id);
event.battleServerAddress = response.instancedMissionBattle;
soundAlarm = true;
}
if (event)
_eventDispatcher.dispatchEvent(event);
}

if (_firstTimeInit)
{
TimeLog.endTimeLog(TimeLogEnum.SERVER_GAME_DATA, "starbase");
var serverEvent:ServerEvent = new ServerEvent(ServerEvent.AUTHORIZED);
_eventDispatcher.dispatchEvent(serverEvent);
```

```
_firstTimeInit = soundAlarm = false;
}

//play sound
if (soundAlarm)
_soundController.playSound(AudioEnum.AFX_GLOBAL_ALARM);

_fleetModel.updateFleet(null);
}

public function starbaseBattleAlert( response:StarbaseBattleAlertResponse ):void
{
//see if any fleets are in battle
var fleets:Vector.<FleetVO> = _fleetModel.fleets;
var fleetVO:FleetVO;
var inBattle:Boolean = Application.STATE == null || ((Application.STATE ==
StateEvent.GAME_BATTLE_INIT || Application.STATE == StateEvent.GAME_BATTLE) &&
!_battleModel.finished);
var newBaseCombat:Boolean = false;
var soundAlarm:Boolean = false;
var notifyFleetID:String;
for (var i:int = 0; i < fleets.length; i++)
{
fleetVO = fleets[i];
if (response.fleetBattles[fleetVO.id])
{
//if this is a new battle then we want to notify. also notify if the fleet we are currently watching
comes under attack from a different opponent
if ((!fleetVO.inBattle && !inBattle) || (_battleModel.battleServerAddress ==
fleetVO.battleServerAddress && response.fleetBattles[fleetVO.id] !=
fleetVO.battleServerAddress))
notifyFleetID = fleetVO.id;
if (!fleetVO.inBattle)
soundAlarm = true;
fleetVO.inBattle = true;
fleetVO.battleServerAddress = response.fleetBattles[fleetVO.id];
} else
{
fleetVO.inBattle = false;
fleetVO.battleServerAddress = null;
}
//show or hide the battle alert if we're in the sector view
if (Application.STATE == StateEvent.GAME_SECTOR)
{
if (SectorInteractSystem(_game.getSystem(SectorInteractSystem)).selected &&
SectorInteractSystem(_game.getSystem(SectorInteractSystem)).selected.id == fleetVO.id)
{
ISectorPresenter(_presenter).onBattle();
//don't want to send out the notification if the battle alert is going to show
if
```

```
(notifyFleetID == fleetVO.id)
notifyFleetID = null;
if (_inFTE)
{
_inFTE = false;
_fteController.nextStep();
}
}
}
}

//set the battle server address of center space battles
/*if (_starbaseModel.centerSpaceBase)
{
if (response.centerSpaceBaseBattle != null && response.centerSpaceBaseBattle != "")
{
if (_starbaseModel.centerSpaceBase.battleServerAddress == null)
soundAlarm = true;
_starbaseModel.centerSpaceBase.battleServerAddress = response.centerSpaceBaseBattle;
} else
_starbaseModel.centerSpaceBase.battleServerAddress = null;
}*/
//set the battle server address of homebase battles
if (response.homeBaseBattle != null && response.homeBaseBattle != "")
{
if (_starbaseModel.homeBase.battleServerAddress == null)
{
newBaseCombat = true;
soundAlarm = true;
}
_starbaseModel.homeBase.battleServerAddress = response.homeBaseBattle;
} else
_starbaseModel.homeBase.battleServerAddress = null;

if (!_fteController.running && Application.STATE != null && !inBattle)
{
var event:StarbaseEvent;
if (newBaseCombat && response.centerSpaceBaseBattle && response.centerSpaceBaseBattle
!= "" && response.centerSpaceBaseBattle != "0")
{
//notify of a center space battle
event = new StarbaseEvent(StarbaseEvent.ALERT_STARBASE_BATTLE,
_starbaseModel.centerSpaceBase.id);
event.battleServerAddress = response.centerSpaceBaseBattle;
soundAlarm = true;
} else if (newBaseCombat && response.homeBaseBattle && response.homeBaseBattle != "")
{
//notify of a home base battle
event = new StarbaseEvent(StarbaseEvent.ALERT_STARBASE_BATTLE,
_starbaseModel.homeBase.id);
event.battleServerAddress
```

```
= response.homeBaseBattle;
soundAlarm = true;
} else if (notifyFleetID)
{
//notify that a fleet is in battle
event = new StarbaseEvent(StarbaseEvent.ALERT_FLEET_BATTLE, null);
event.fleetID = notifyFleetID;
event.battleServerAddress = response.fleetBattles[notifyFleetID];
}
if (event)
_eventDispatcher.dispatchEvent(event);
}

if (_firstTimeInit)
{
TimeLog.endTimeLog(TimeLogEnum.SERVER_GAME_DATA, "starbase");
var serverEvent:ServerEvent = new ServerEvent(ServerEvent.AUTHORIZED);
_eventDispatcher.dispatchEvent(serverEvent);
_firstTimeInit = soundAlarm = false;
}

//play sound
if (soundAlarm)
_soundController.playSound(AudioEnum.AFX_GLOBAL_ALARM);

_fleetModel.updateFleet(null);
}

/**
*
* @param response
*/
public function handleStarbaseMissionCompleteResponse(
response:StarbaseMissionCompleteResponse ):void
{
if (Application.STATE != null)
{
var mission:MissionVO = _missionModel.getMissionByID(response.missionPersistence);
if (mission && mission.isFTE)
{
if (mission.getValue("automaticallyAcceptRewards") == false)
_transactionController.missionAcceptRewards(mission.id);
_fteController.checkMissionRequired(mission);
} else if (mission && !mission.complete && mission.category != MissionEnum.DAILY)
{
_missionModel.missionComplete();
_transactionController.dataImported();
}
}
}

/**
```

```
*
* @param response
*/
public function handleStarbaseFleetDockedResponse(
response:StarbaseFleetDockedResponse ):void
{
if (response.alloyCargo + response.energyCargo + response.syntheticCargo > 0)
{
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = ToastEnum.FLEET_DOCKED;
toastEvent.addStrings(_fleetModel.getFleet(response.fleetPersistence).name, _hasDocked,
response.alloyCargo, _alreadyRecieved, response.energyCargo, response.syntheticCargo);
_eventDispatcher.dispatchEvent(toastEvent);
}
}

/**
*
* @param response
*/
public function handleStarbaseBountyRewardResponse(
response:StarbaseBountyRewardResponse ):void
{
/*var toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = ToastEnum.BOUNTY_REWARD;
toastEvent.addStrings(response.bounty);
_eventDispatcher.dispatchEvent(toastEvent);*/
}

public function handleMessageOftheDayResponse( response:StarbaseMotdListResponse ):void
{
_motdModel.addMessages(response.motds);
}

public function requestMotDRead( motdKey:String ):void
{
var motDReadRequest:StarbaseMotDReadRequest =
StarbaseMotDReadRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MARK_MOTD_READ_MESSAGE));
motDReadRequest.key = motdKey;
_serverController.send(motDReadRequest);
}

public function handleStarbaseDailyResponse( response:StarbaseDailyResponse ):void
{
_motdDailyModel.addData(response.escalation, response.canNextClaimDelta,
response.dailyResetsDelta, response.header, response.protocolID);
}

public
```

```
function handleStarbaseDailyRewardResponse( response:StarbaseDailyRewardResponse
):void
{
_motdDailyModel.addRewardData(response);
}
public function requestDailyClaim( header:int, protocolID:int ):void
{
var motDDailyClaimRequest:StarbaseClaimDailyRequest =
StarbaseClaimDailyRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_CLAIM_DAILY_MESSAGE));
_serverController.send(motDDailyClaimRequest);
}

public function handleStarbaseAvailableRerollResponse(
response:StarbaseAvailableRerollResponse ):void
{
_battleModel.addAvailableReroll(response);
}
public function handleStarbaseUnavailableRerollResponse(
response:StarbaseUnavailableRerollResponse ):void
{
_battleModel.addUnavailableReroll(response);
}
public function handleStarbaseRerollChanceResponse(
response:StarbaseRerollChanceResultResponse ):void
{
_battleModel.updateRerollFromScan(response);
}
public function handleStarbaseRerollReceivedResponse(
response:StarbaseRerollReceivedResultResponse ):void
{
_battleModel.updateRerollFromReroll(response);
}

public function handleStarbaseMoveStarbaseResponse(
response:StarbaseMoveStarbaseResponse ):void
{
if (response.status == 0)
{
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = ToastEnum.BASE_RELOCATED;
toastEvent.addStrings(_baseRelocatedTitle, _baseRelocatedBody);
_eventDispatcher.dispatchEvent(toastEvent);
}

}


//======================================================================================
//**************************************************************************************
//
```

```
ACHIEVEMENTS
//**************************************************************************************************

//==========================================================================

public function handleStarbaseAchievementsResponse(
response:StarbaseAchievementsResponse ):void
{
_achievementModel.addData(response);
}

public function requestAchievements():void
{
var requestAchievements:StarbaseRequestAchievementsRequest =
StarbaseRequestAchievementsRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIEN
RequestEnum.STARBASE_REQUEST_ACHIEVEMENTS));
_serverController.send(requestAchievements);
}

public function handleStarbaseAllScoresResponse( response:StarbaseAllScoresResponse
):void
{
_achievementModel.addAllScoreData(response);
}

public function requestAllScores():void
{
var requestAllScores:StarbaseAllScoresRequest =
StarbaseAllScoresRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_REQUEST_ALL_SCORES));
_serverController.send(requestAllScores);
}

public function claimAchievementReward( achievement:String ):void
{
var starbaseClaimAchievementRewardRequest:StarbaseClaimAchievementRewardRequest =
StarbaseClaimAchievementRewardRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CL
RequestEnum.
STARBASE_CLAIM_ACHIEVEMENT_REWARD));
starbaseClaimAchievementRewardRequest.achievement = achievement;
_serverController.send(starbaseClaimAchievementRewardRequest);
}

public function mintNFT( tokenType:int, tokenAmount:int, tokenPrototype:String ):void
{
_transactionController.mintNFTTransaction(tokenType, tokenAmount, tokenPrototype);
}


//==========================================================================
//**************************************************************************************************
```

```
// PAYMENTS
//**************************************************************************************************

//=========================================================================

public function handleStarbaseGetPaywallPayoutsResponse(
response:StarbaseGetPaywallPayoutsResponse ):void
{
var paywall:PaywallEvent = new PaywallEvent(PaywallEvent.OPEN_PAYWALL);
paywall.paywallData = response.data;
_eventDispatcher.dispatchEvent(paywall);
}

public function requestPaywallPayouts():void
{
if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
var paywall:PaywallEvent = new PaywallEvent(PaywallEvent.OPEN_PAYWALL);
paywall.paywallData = ExternalInterfaceAPI.GetFacebookItems();
_eventDispatcher.dispatchEvent(paywall);
}
else
{
//ExternalInterfaceAPI.logConsole("Open Kongregate Payment 2");
var requestPayouts:StarbaseGetPaywallPayoutsRequest =
StarbaseGetPaywallPayoutsRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_GET_PAYWALL_PAYOUTS));
_serverController.send(requestPayouts);
}
}

public function requestPaymentVerification( externalTrkid:String = '', payoutId:String = '',
responseData:String = '', responseSignature:String = '' ):void
{
var req:StarbaseVerifyPaymentRequest =
StarbaseVerifyPaymentRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_VERIFY_PAYMENT));
req.externalTrkid = externalTrkid;
req.payoutId = payoutId;
req.responseData = responseData;
req.responseSignature = responseSignature;
_serverController.send(req);
_logger.debug('requestPaymentVerification - Request = {0}, externalTrkid = {1}, payoutId = {2},
responseData = {3}, responseSignature = {4}', [req, externalTrkid, payoutId, responseData,
responseSignature]);
}



//=========================================================================
```

```
//**************************************************************************************************************
// MAIL
//**************************************************************************************************************

//==========================================================================================
/**
*
* @param response
*/
public function handleMailInboxResponse( response:MailInboxResponse ):void
{
_mailModel.addMailHeaders(response.mailData);
var mailInvites:Dictionary = _mailModel.getMailInvites();
_allianceModel.setEmailInvites(mailInvites);
// Load alliances required to set up invitaions.
var modelAlliances:Dictionary = _allianceModel.getAlliances();
for (var key:Object in mailInvites)
{
if (!(mailInvites[key] in modelAlliances))
{
var getAllianceBaseline:AllianceRequestBaselineRequest =
AllianceRequestBaselineRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_REQUEST_BASELINE));
getAllianceBaseline.allianceKey = mailInvites[key];
_serverController.send(getAllianceBaseline);
}
}
}

public function handleMailDetailResponse( response:MailDetailResponse ):void
{
_mailModel.addMailDetail(response.key, response.sender, response.senderAlliance,
response.body, response.senderRace, response.html);
}

public function handleUnreadResponse( response:MailUnreadResponse ):void
{
_mailModel.updateCount(response.unread, response.total, true);
}

public function mailSendMessage( playerID:String, subject:String, body:String ):void
{
var sendMessage:MailSendMailRequest =
MailSendMailRequest(_serverController.getRequest(ProtocolEnum.MAIL_CLIENT,
RequestEnum.MAIL_SEND_MAIL));
sendMessage.recipient = playerID;
sendMessage.subject = subject;
sendMessage.body = body;
_serverController.send(sendMessage);
}
```

```
public function mailSendAllianceMessage( subject:String, body:String ):void
{
var sendMessage:MailSendAllianceMailRequest =
MailSendAllianceMailRequest(_serverController.getRequest(ProtocolEnum.MAIL_CLIENT,
RequestEnum.MAIL_SEND_ALLIANCE_MAIL));
sendMessage.subject = subject;
sendMessage.body = body;
_serverController.send(sendMessage);
}

public function mailGetMailbox():void
{
var getMail:MailRequestInboxRequest =
MailRequestInboxRequest(_serverController.getRequest(ProtocolEnum.MAIL_CLIENT,
RequestEnum.MAIL_REQUEST_INBOX));
_serverController.send(getMail);
}

public function mailGetMailDetail( mailKey:String ):void
{
var getMailDetail:MailReadMailRequest =
MailReadMailRequest(_serverController.getRequest(ProtocolEnum.MAIL_CLIENT,
RequestEnum.MAIL_READ_MAIL));
getMailDetail.mail = mailKey;
_serverController.send(getMailDetail);
}

public function mailDelete( v:Vector.<String> ):void
{
var mailDelete:MailDeleteMailRequest =
MailDeleteMailRequest(_serverController.getRequest(ProtocolEnum.MAIL_CLIENT,
RequestEnum.MAIL_DELETE_MAIL));
mailDelete.mail = v;
_serverController.send(mailDelete);
}


//==========================================================================
//**************************************************************************
// BATTLE LOG (starbase)
//**************************************************************************

//==========================================================================

public function battleLogGetBattleListStarbase():void
{
var getBattleLogList:BattleLogListRequest =
BattleLogListRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BATTLELOG_LIST));
```

```
_serverController.send(getBattleLogList);
}

public function battleLogGetBattleDetailStarbase( battleLogID:String ):void
{
var getBattleLogDetail:BattleLogDetailRequest =
BattleLogDetailRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BATTLELOG_DETAILS));
getBattleLogDetail.battleLogID = battleLogID;
_serverController.send(getBattleLogDetail);
}

public function handleStarbaseBattleLogListResponse( response:BattleLogListResponse ):void
{
_battleLogModel.addBattleLogList(response.battles);
}

public function handleStarbaseBattleLogDetailsResponse( response:BattleLogDetailsResponse
):void
{
_battleLogModel.addBattleLogDetail(response);
}



//==========================================================================================
//******************************************************************************************
// BATTLE LOG (Mongo)
//******************************************************************************************

//==========================================================================================

public function battleLogGetBattleList( filter:String ):void
{
if (Application.BATTLE_WEB_PATH == null)
{
battleLogGetBattleListStarbase();
return;
}

var absoluteUrl:String = Application.BATTLE_WEB_PATH + "?p=" + CurrentUser.id;
switch (filter)
{
case BattleLogFilterEnum.SELFPVP:
absoluteUrl += "&pvp=true";
break;
case BattleLogFilterEnum.SELFPVE:
absoluteUrl += "&pve=true";
break;
case
```

```
BattleLogFilterEnum.BASEPVP:
absoluteUrl = Application.BATTLE_WEB_PATH + "?pvpbase";
break;
case BattleLogFilterEnum.FLEETPVP:
absoluteUrl = Application.BATTLE_WEB_PATH + "?pvpfleet";
break;
case BattleLogFilterEnum.BESTPVE:
absoluteUrl = Application.BATTLE_WEB_PATH + "?bestpve";
break;
case BattleLogFilterEnum.SELFALL:
default:
break;
}

trace("requesting " + absoluteUrl);
AssetModel.instance.getFromCache(absoluteUrl, handleBattleLogListResponse,
LoadPriority.HIGH, true);
}

public function battleLogGetBattleDetail( battleId:String ):void
{
if (Application.BATTLE_WEB_PATH == null)
{
battleLogGetBattleDetailStarbase(battleId)
return;
}
var absoluteUrl:String = Application.BATTLE_WEB_PATH + "?detail=" + battleId;
trace("requesting " + absoluteUrl);
AssetModel.instance.getFromCache(absoluteUrl, handleBattleLogDetailsResponse,
LoadPriority.HIGH, true);
}

public function handleBattleLogListResponse( json:Object ):void
{
var absoluteUrl:String = Application.BATTLE_WEB_PATH + "?p=" + CurrentUser.id;
AssetModel.instance.removeFromCache(absoluteUrl);
if (json == AssetModel.FAILED)
{
trace("failed", absoluteUrl);
return;
}

trace("received", absoluteUrl);
var response:BattleLogListResponse = new BattleLogListResponse();
response.readJSON(json);
_battleLogModel.addBattleLogList(response.battles);
}

public function handleBattleLogDetailsResponse( json:Object ):void
{
if
```

```
(json == AssetModel.FAILED)
{
return;
}
trace("received BattleLogDetailsResponse");
var response:BattleLogDetailsResponse = new BattleLogDetailsResponse();
response.readJSON(json);
_battleLogModel.addBattleLogDetail(response);
}


//==========================================================================
//**************************************************************************************************
// CHAT
//**************************************************************************************************

//==========================================================================

public function handleChatBaselineResponse( response:ChatBaselineResponse ):void
{
_chatController.addBlockedUsers(response);
}

public function requestReportPlayer( playerID:String ):void
{
var reportPlayerRequest:ChatReportChatRequest =
ChatReportChatRequest(_serverController.getRequest(ProtocolEnum.CHAT_CLIENT,
RequestEnum.CHAT_REPORT_CHAT));
reportPlayerRequest.playerKey = playerID;
_serverController.send(reportPlayerRequest);
}



//==========================================================================
//**************************************************************************************************
// BOOKMARKS
//**************************************************************************************************

//==========================================================================

public function bookmarkSave( name:String, sector:String, nameProto:String, enumProto:String,
sectorProto:String, x:int, y:int ):void
{
var saveBookmark:StarbaseBookmarkSaveRequest =
StarbaseBookmarkSaveRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BOOKMARK_SAVE));
saveBookmark.name = name;
saveBookmark.sector = sector;
saveBookmark.nameProto = nameProto;
saveBookmark.enumProto
```

```
= enumProto;
saveBookmark.sectorProto = sectorProto;
saveBookmark.x = x;
saveBookmark.y = y;
_serverController.send(saveBookmark);
}

public function bookmarkUpdate( bookmark:BookmarkVO ):void
{
var updateBookmark:StarbaseBookmarkUpdateRequest =
StarbaseBookmarkUpdateRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BOOKMARK_UPDATE));
updateBookmark.bookmark = bookmark;
_serverController.send(updateBookmark);
}

public function bookmarkDelete( bookmarkIndex:uint ):void
{
var deleteBookmark:StarbaseBookmarkDeleteRequest =
StarbaseBookmarkDeleteRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BOOKMARK_DELETE));
deleteBookmark.index = bookmarkIndex;
_serverController.send(deleteBookmark);
}


//==========================================================================
//**************************************************************************************************
// LEADERBOARD
//**************************************************************************************************

//==========================================================================

public function handleWarfrontUpdate( response:WarfrontUpdateResponse ):void
{
_warfrontModel.importData(response);
}

public function leaderboardRequest( type:int, scope:int ):void
{
var leaderboardRequest:LeaderboardRequest =
LeaderboardRequest(_serverController.getRequest(ProtocolEnum.LEADERBOARD_CLIENT,
RequestEnum.LEADERBOARD_REQUEST_LEADERBOARD));
leaderboardRequest.type = type;
leaderboardRequest.scope = scope;
_serverController.send(leaderboardRequest);
}

public function leaderboardRequestPlayerProfile( playerKey:String, name:String ):void
{
var
```

```
leaderboardRequestPlayerProfile:LeaderboardRequestPlayerProfileRequest =
LeaderboardRequestPlayerProfileRequest(_serverController.getRequest(ProtocolEnum.LEADERBOARD_
RequestEnum.
LEADERBOARD_REQUEST_PLAYER_PROFILE));
leaderboardRequestPlayerProfile.playerKey = playerKey;
leaderboardRequestPlayerProfile.nameSearch = name;
_serverController.send(leaderboardRequestPlayerProfile);
}

public function handlePlayerProfile( response:PlayerProfileResponse ):void
{
_playerModel.addPlayers(response.players)
}

public function handleLeaderboardUpdate( response:LeaderboardResponse ):void
{
_leaderboardModel.updateLeaderboardEntry(response);
}



//==========================================================================================
//******************************************************************************************
// OFFER
//******************************************************************************************

//==========================================================================================

public function handleStarbaseOfferRedeemed( response:StarbaseOfferRedeemedResponse
):void
{
CurrentUser.removeOffer(response.offerPrototype);
}


//==========================================================================================
//******************************************************************************************
// ALLIANCE
//******************************************************************************************

//==========================================================================================

public function allianceSendInvite( playerKey:String ):void
{
var sendInvite:AllianceSendInviteRequest =
AllianceSendInviteRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SEND_INVITE));
sendInvite.playerKey = playerKey;
_serverController.send(sendInvite);
}

public
```

```
function handleAllianceBaselineResponse( response:AllianceBaselineResponse ):void
{
_allianceModel.addAlliance(response.alliance);
_allianceModel.addEmailInvites();
}

public function handleAllianceRosterResponse( response:AllianceRosterResponse ):void
{
_allianceModel.updateMembers(response.allianceKey, response.members);
}

public function handleAlliancePublicResponse( response:PublicAlliancesResponse ):void
{
_allianceModel.addOpenAlliances(response.alliances);
}

public function handleAllianceGenericResponse( response:AllianceGenericResponse ):void
{
_allianceModel.handleGenericMessage(response.responseEnum, response.allianceKey)
}

public function handleAllianceInviteResponse( response:AllianceInviteResponse ):void
{
_allianceModel.addInvitedAlliance(response.inviteVO)
}


//==========================================================================
//**************************************************************************************************
// GENERAL
//**************************************************************************************************

//==========================================================================

/**
*
* @param id
* @param targetID
* @param moduleID
*/
public function toggleModule( id:String, targetID:String, moduleID:int ):void
{
var order:BattleToggleModuleOrderRequest =
BattleToggleModuleOrderRequest(_serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_TOGGLE_MODULE_ORDER));
order.entityID = id;
order.targetID = targetID; // TODO - does this make sense in this message?
order.issuedTick = ServerController.SIMULATED_TICK;
order.moduleID = moduleID;
_serverController.send(order);
}
```

```
private function destroyEntity( entityKey:String, reason:int ):void
{
//remove entities
var entity:Entity = _game.getEntity(entityKey);
if (entity)
{
var detail:Detail = entity.get(Detail);
var position:Position = entity.get(Position);
if (reason == RemoveReasonEnum.Destroyed)
{
if (Application.STATE == StateEvent.GAME_SECTOR)
_vfxFactory.createSectorExplosion(entity, position.x, position.y);
else
_vfxFactory.createExplosion(entity, position.x, position.y);
}

switch (detail.category)
{
case CategoryEnum.SHIP:
if (entity.get(Ship))
{
var modules:Modules = entity.get(Modules);
for (var j:int = 0; j < modules.entityModules.length; j++)
{
if (modules.entityModules[j])
_attackFactory.destroyAttack(modules.entityModules[j]);
}
_shipFactory.destroyShip(entity);

} else
{
// update the fleet state if it's a fleet
if (entity.get(Owned))
{
var fleetVO:FleetVO = _fleetModel.getFleet(entity.id);
if (reason == RemoveReasonEnum.Docked)
fleetVO.state = FleetStateEnum.DOCKED;
}
if (reason == RemoveReasonEnum.TransgateTravel && Animation(entity.get(Animation)).visible
== true)
{
//find the transgate the player entered
var entities:Array = GridSystem(_game.getSystem(GridSystem)).getEntitiesAt(position.x,
position.y);
for (var i:int = 0; i < entities.length; i++)
{
if (entities[i].entity.has(Transgate))
{
//play
```

```
the transgate animation
if (!entities[i].animation.playing)
entities[i].animation.playing = true;
}
}
_soundController.playSound(AudioEnum.AFX_STG_TRANSGATE_ACTIVATION, 0.5);
}
_shipFactory.destroyFleet(entity);
}
break;
case CategoryEnum.BUILDING:
var health:Health = Health(entity.get(Health));
//set the health of the building to 0 instead of removing it so the destroyed state is shown
if (health)
health.currentHealth = 0;
if (entity.has(Interactable))
ObjectPool.give(entity.remove(Interactable));
if (entity.has(DebuffTray))
{
var vcList:VCList = entity.get(VCList);
vcList.removeComponentType(TypeEnum.DEBUFF_TRAY);
ObjectPool.give(entity.remove(DebuffTray));
}
if (detail.prototypeVO && detail.prototypeVO.itemClass == TypeEnum.FORCEFIELD)
_starbaseFactory.destroyStarbaseItem(entity);
break;
case CategoryEnum.STARBASE:
_starbaseFactory.destroyStarbaseItem(entity);
break;
case CategoryEnum.SECTOR:
_sectorFactory.destroySectorEntity(entity);
break;
}
}
}

private function fireDrones( drones:Array ):void
{
var ship:Entity;
for (var i:int = 0; i < drones.length; i++)
{
var attack:DroneAttackData = drones[i];
if (!_game.getEntity(attack.attackId))
{
ship = _game.getEntity(attack.entityOwnerId);
_attackFactory.createDrone(ship, attack);
}
}
}

private
```

```
function fireProjectiles( projectiles:Array ):void
{
var ship:Entity;
for (var i:int = 0; i < projectiles.length; i++)
{
var attack:ProjectileAttackData = projectiles[i];
if (!_game.getEntity(attack.attackId))
{
ship = _game.getEntity(attack.entityOwnerId);
_attackFactory.createProjectile(ship, attack);
}
}
}

private function fireAreaAttacks( areaAttacks:Array ):void
{
var ship:Entity;
for (var i:int = 0; i < areaAttacks.length; i++)
{
var attack:AreaAttackData = areaAttacks[i];

if (!_game.getEntity(attack.attackId))
{
ship = _game.getEntity(attack.entityOwnerId);
_attackFactory.createArea(ship ? ship.id : '', attack);
}
}
}

private function fireBeams( beams:Array ):void
{
for (var i:int = 0; i < beams.length; i++)
{
var attack:BeamAttackData = beams[i];
if (!_game.getEntity(attack.attackId))
_attackFactory.createBeam(attack);
}
}

private function battleAddPlayers( players:Array ):void
{
for (var i:int = 0; i < players.length; i++)
{
_playerModel.addPlayer(PlayerVO(players[i]));
}
}

private function addPlayers( players:Vector.<PlayerVO> ):void
{
for (var i:int = 0; i < players.length; i++)
{
```

```
_playerModel.addPlayer(players[i]);
}
}

private function battleRemovePlayers( players:Array ):void
{
for (var i:int = 0; i < players.length; i++)
{
_playerModel.removePlayer(RemovedObjectData(players[i]).id);
}
}

private function removePlayers( players:Vector.<RemovedObjectData> ):void
{
for (var i:int = 0; i < players.length; i++)
{
_playerModel.removePlayer(players[i].id);
}
}

private function globalErrorHandler( errStr:String ):void
{
//if(CONFIG::DEBUG == true)
{
// @todo: throttle
var msg:ProxyReportCrashRequest =
ProxyReportCrashRequest(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_REPORT_CRASH));
msg.dataStr = errStr;

_serverController.send(msg);
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
var nClientCrashView:ClientCrashView =
ClientCrashView(_viewFactory.createView(ClientCrashView));
nClientCrashView.errorMsg = errStr;
viewEvent.targetView = nClientCrashView;
_eventDispatcher.dispatchEvent(viewEvent);
}
}

public function disconnect():void { _firstTimeInit = true; }

/**
*
* @param serverController
*/
public function give( serverController:ServerController ):void
{
_serverController = serverController;
_transactionController.serverController
```

```
= _serverController;
_fteController.serverController = _serverController;
_tickProvider.addFrameListener(onTick);
}

public function set inFTE( v:Boolean ):void { _inFTE = v; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set attackFactory( v:IAttackFactory ):void { _attackFactory = v; }
[Inject]
public function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
[Inject]
public function set fleetModel( value:FleetModel ):void { _fleetModel = value; }
[Inject]
public function set fteController( v:FTEController ):void { _fteController = v; }
[Inject]
public function set missionModel( value:MissionModel ):void { _missionModel = value; }
[Inject]
public function set mailModel( v:MailModel ):void { _mailModel = v; }
[Inject]
public function set blueprintModel( value:BlueprintModel ):void { _blueprintModel = value; }
[Inject]
public function set playerModel( value:PlayerModel ):void { _playerModel = value; }
public function get presenter():IGamePresenter { return _presenter; }
public function set presenter( v:IGamePresenter ):void { _presenter = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set sectorFactory( v:ISectorFactory ):void { _sectorFactory = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set shipFactory( v:IShipFactory ):void { _shipFactory = v; }
[Inject]
public function set soundController( v:SoundController ):void { _soundController = v; }
[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set battleLogModel( v:BattleLogModel ):void { _battleLogModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set settingsController( v:SettingsController ):void { _settingsController = v; }
[Inject]
```

```
public function set chatController( v:ChatController ):void { _chatController = v; }
[Inject]
public function set eventController( v:EventController ):void { _eventController = v; }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }
[Inject]
public function set viewFactory( v:IViewFactory ):void { _viewFactory = v; }
[Inject]
public function set viewStack( v:IViewStack ):void { _viewStack = v; }
[Inject]
public function set warfrontModel( v:WarfrontModel ):void { _warfrontModel = v; }
[Inject]
public function set leaderboardModel( v:LeaderboardModel ):void { _leaderboardModel = v; }
[Inject]
public function set allianceModel( v:AllianceModel ):void { _allianceModel = v; }
[Inject]
public function set motdModel( v:MotDModel ):void { _motdModel = v; }
[Inject]
public function set motdDailyModel( v:MotDDailyRewardModel ):void { _motdDailyModel = v; }
[Inject]
public function set achievementModel( v:AchievementModel ):void { _achievementModel = v; }
}
}

-------------------------------------
File 7: igw\com\controller\ServerController.as
package com.controller
{
import com.Application;
import com.enum.TimeLogEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.enum.server.ResponseEnum;
import com.event.ServerEvent;
import com.model.asset.AssetModel;
import com.service.loading.LoadPriority;
import com.service.server.EightTrack;
import com.service.server.IRequest;
import com.service.server.IResponse;
import com.service.server.ITickedResponse;
import com.service.server.ITransactionResponse;
import com.service.server.PacketFactory;
import com.service.server.TinCan;
import com.service.server.incoming.alliance.AllianceBaselineResponse;
import com.service.server.incoming.alliance.AllianceGenericResponse;
import com.service.server.incoming.alliance.AllianceInviteResponse;
import com.service.server.incoming.alliance.AllianceRosterResponse;
import com.service.server.incoming.alliance.PublicAlliancesResponse;
import com.service.server.incoming.battle.BattleDataResponse;
import
```

```
com.service.server.incoming.battle.BattleDebugLinesResponse;
import com.service.server.incoming.battle.BattleHasEndedResponse;
import com.service.server.incoming.battlelog.BattleLogDetailsResponse;
import com.service.server.incoming.battlelog.BattleLogListResponse;
import com.service.server.incoming.chat.ChatBaselineResponse;
import com.service.server.incoming.chat.ChatResponse;
import com.service.server.incoming.leaderboard.LeaderboardResponse;
import com.service.server.incoming.leaderboard.PlayerProfileResponse;
import com.service.server.incoming.leaderboard.WarfrontUpdateResponse;
import com.service.server.incoming.mail.MailDetailResponse;
import com.service.server.incoming.mail.MailInboxResponse;
import com.service.server.incoming.mail.MailUnreadResponse;
import com.service.server.incoming.proxy.ProxyBattleDisconnectedResponse;
import com.service.server.incoming.proxy.ProxySectorDisconnectedResponse;
import com.service.server.incoming.proxy.ProxyStarbaseDisconnectedResponse;
import com.service.server.incoming.sector.SectorAlwaysVisibleBaselineResponse;
import com.service.server.incoming.sector.SectorAlwaysVisibleUpdateResponse;
import com.service.server.incoming.sector.SectorBaselineResponse;
import com.service.server.incoming.sector.SectorFleetTravelAlertResponse;
import com.service.server.incoming.sector.SectorUpdateResponse;
import com.service.server.incoming.starbase.StarbaseAchievementsResponse;
import com.service.server.incoming.starbase.StarbaseAllScoresResponse;
import com.service.server.incoming.starbase.StarbaseAvailableRerollResponse;
import com.service.server.incoming.starbase.StarbaseBaselineResponse;
import com.service.server.incoming.starbase.StarbaseBattleAlertResponse;
import com.service.server.incoming.starbase.StarbaseBountyRewardResponse;
import com.service.server.incoming.starbase.StarbaseDailyResponse;
import com.service.server.incoming.starbase.StarbaseDailyRewardResponse;
import com.service.server.incoming.starbase.StarbaseFleetDockedResponse;
import com.service.server.incoming.starbase.StarbaseGetPaywallPayoutsResponse;
import com.service.server.incoming.starbase.StarbaseInstancedMissionAlertResponse;
import com.service.server.incoming.starbase.StarbaseMissionCompleteResponse;
import com.service.server.incoming.starbase.StarbaseMotdListResponse;
import com.service.server.incoming.starbase.StarbaseMoveStarbaseResponse;
import com.service.server.incoming.starbase.StarbaseOfferRedeemedResponse;
import com.service.server.incoming.starbase.StarbaseRerollChanceResultResponse;
import com.service.server.incoming.starbase.StarbaseRerollReceivedResultResponse;
import com.service.server.incoming.starbase.StarbaseUnavailableRerollResponse;
import com.service.server.incoming.universe.UniverseNeedCharacterCreateResponse;
import com.service.server.incoming.universe.UniverseSectorListResponse;
import com.service.server.outgoing.proxy.ProxyConnectToBattleRequest;
import com.service.server.outgoing.proxy.TimeSyncRequest;
import com.ui.modal.server.DisconnectedView;
import com.util.TimeLog;

import flash.events.IEventDispatcher;
import flash.events.TimerEvent;
import flash.utils.ByteArray;
import flash.utils.CompressionAlgorithm;
import flash.utils.Endian;
import
```

```
flash.utils.Timer;

import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;

import com.service.ExternalInterfaceAPI;

public class ServerController
{
public static var INTERPOLATION:Number = 0;
public static var SERVER_TICK:int = 0;
public static var SIMULATED_TICK:int = 100;
public static var TIME_STEP:int = 0;
public static const TARGET_TIME_STEP:int = 100;

private var _chatController:ChatController;
private var _eventDispatcher:IEventDispatcher;
private var _gameController:GameController;
private var _settingsController:SettingsController;
private var _tickedResponses:Vector.<ITickedResponse>;
private var _viewFactory:IViewFactory;
private var _temp:Number;
private var _tempInterp:Number;
private var _ticks:Array;
private var _time:Number;
private var _t:Number;
private var _proxy:TinCan;
private var _replayDecoder:EightTrack;
private var _keepAliveTimer:Timer;


[PostConstruct]
public function init():void
{
_proxy = new TinCan();
_tickedResponses = new Vector.<ITickedResponse>;
_ticks = [];
_time = 0;

_keepAliveTimer = new Timer(60000);
_keepAliveTimer.addEventListener(TimerEvent.TIMER, onKeepAliveTick, false, 0, true);
TimeLog.serverController = this;

_gameController.give(this);
_chatController.give(this);
}

public function connect( ip:String, port:int, policy:String, devConnection:Boolean = false ):void
{
//ExternalInterfaceAPI.logConsole("Imperium IP proxy = " + ip + " , port = " + port.toString());
TimeLog.startTimeLog(TimeLogEnum.CONNECT_TO_PROXY);
```

```
//connect to the game's proxy server
_proxy.init(ip, port, policy, TinCan.GAME, devConnection);
_proxy.addResponseListener(handleResponse);
_proxy.addConnectionListener(onProxyConnect);
_proxy.serverController = this;

_settingsController.giveServerController(this);
}

private function onKeepAliveTick( e:TimerEvent ):void
{
var keepAliveTick:TimeSyncRequest =
TimeSyncRequest(getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_TIME_SYNC));
_proxy.send(keepAliveTick);
}

private function handleResponse( response:IResponse ):void
{
//ExternalInterfaceAPI.logConsole("Imperium - response handling");
switch (response.protocolID)
{
case ProtocolEnum.PROXY_CLIENT:
switch (response.header)
{
case ResponseEnum.AUTHORIZATION:
//ExternalInterfaceAPI.logConsole("Imperium - authorization");
var serverEvent:ServerEvent = new ServerEvent(ServerEvent.AUTHORIZED, response);
_eventDispatcher.dispatchEvent(serverEvent);
break;
case ResponseEnum.PROXY_BATTLE_DISCONNECTED:
//ExternalInterfaceAPI.logConsole("Imperium - dc");

_gameController.handleProxyBattleDisconnected(ProxyBattleDisconnectedResponse(response));
break;
case ResponseEnum.PROXY_SECTOR_DISCONNECTED:
//ExternalInterfaceAPI.logConsole("Imperium - dc 1");

_gameController.handleProxySectorDisconnected(ProxySectorDisconnectedResponse(response));
break;
case ResponseEnum.PROXY_STARBASE_DISCONNECTED:
//ExternalInterfaceAPI.logConsole("Imperium - dc 2");

_gameController.handleProxyStarbaseDisconnected(ProxyStarbaseDisconnectedResponse(response));
break;
}
break;
case ProtocolEnum.SECTOR_CLIENT:
switch (response.header)
{
```

```
case ResponseEnum.SECTOR_ALWAYS_VISIBLE_BASELINE:
//ExternalInterfaceAPI.logConsole("Imperium - visible");

_gameController.handleSectorAlwaysVisibleBaselineResponse(SectorAlwaysVisibleBaselineResponse(re
break;
case ResponseEnum.SECTOR_FLEET_TRAVEL_ALERT:
//ExternalInterfaceAPI.logConsole("Imperium - alert");
_gameController.sectorFleetTravelAlert(SectorFleetTravelAlertResponse(response));
break;
}
break;
case ProtocolEnum.STARBASE_CLIENT:
switch (response.header)
{
case ResponseEnum.STARBASE_TRANSACTION_RESPONSE:
_gameController.handleStarbaseTransactionResponse(ITransactionResponse(response));
break;
case ResponseEnum.STARBASE_BASELINE:
_gameController.handleStarbaseBaselineResponse(StarbaseBaselineResponse(response));
break;
case ResponseEnum.STARBASE_BATTLE_ALERT:
_gameController.starbaseBattleAlert(StarbaseBattleAlertResponse(response));
break;
case ResponseEnum.STARBASE_MISSION_COMPLETE:

_gameController.handleStarbaseMissionCompleteResponse(StarbaseMissionCompleteResponse(respons
break;
case ResponseEnum.STARBASE_FLEET_DOCKED:

_gameController.handleStarbaseFleetDockedResponse(StarbaseFleetDockedResponse(response))
break;
case ResponseEnum.STARBASE_BOUNTY_REWARD:

_gameController.handleStarbaseBountyRewardResponse(StarbaseBountyRewardResponse(response));
break;
case ResponseEnum.STARBASE_BATTLELOG_LIST:
_gameController.handleStarbaseBattleLogListResponse(BattleLogListResponse(response));
break;
case ResponseEnum.STARBASE_BATTLELOG_DETAILS:

_gameController.handleStarbaseBattleLogDetailsResponse(BattleLogDetailsResponse(response));
break;
case ResponseEnum.STARBASE_OFFER_REDEEMED:

_gameController.handleStarbaseOfferRedeemed(StarbaseOfferRedeemedResponse(response));
break;
case ResponseEnum.STARBASE_MOTD_LIST:
_gameController.handleMessageOftheDayResponse(StarbaseMotdListResponse(response));
break;
case
```

```
ResponseEnum.STARBASE_DAILY:
_gameController.handleStarbaseDailyResponse(StarbaseDailyResponse(response));
break;
case ResponseEnum.STARBASE_DAILY_REWARD:

_gameController.handleStarbaseDailyRewardResponse(StarbaseDailyRewardResponse(response));
break;
case ResponseEnum.STARBASE_AVAILABLE_REROLL:

_gameController.handleStarbaseAvailableRerollResponse(StarbaseAvailableRerollResponse(response));
break;
case ResponseEnum.STARBASE_REROLL_CHANCE_RESULT:

_gameController.handleStarbaseRerollChanceResponse(StarbaseRerollChanceResultResponse(respons
break;
case ResponseEnum.STARBASE_REROLL_RECEIVED_RESULT:

_gameController.handleStarbaseRerollReceivedResponse(StarbaseRerollReceivedResultResponse(respo
break;
case ResponseEnum.STARBASE_MOVE_STARBASE_RESPONSE:

_gameController.handleStarbaseMoveStarbaseResponse(StarbaseMoveStarbaseResponse(response));
break;
case ResponseEnum.STARBASE_ACHIEVEMENTS_RESPONSE:

_gameController.handleStarbaseAchievementsResponse(StarbaseAchievementsResponse(response));
break;
case ResponseEnum.STARBASE_ALL_SCORES_RESPONSE:
_gameController.handleStarbaseAllScoresResponse(StarbaseAllScoresResponse(response));
break;
case ResponseEnum.STARBASE_GET_PAYWALL_PAYOUTS_RESPONSE:

_gameController.handleStarbaseGetPaywallPayoutsResponse(StarbaseGetPaywallPayoutsResponse(res
break;
case ResponseEnum.STARBASE_UNAVAILABLE_REROLL:

_gameController.handleStarbaseUnavailableRerollResponse(StarbaseUnavailableRerollResponse(respor
break;
case ResponseEnum.STARBASE_INSTANCED_MISSION_ALERT:

_gameController.starbaseInstancedMissionAlert(StarbaseInstancedMissionAlertResponse(response));
break;
}
break;
case ProtocolEnum.MAIL_CLIENT:
switch (response.header)
{
case ResponseEnum.MAIL_UNREAD:
_gameController.handleUnreadResponse(MailUnreadResponse(response));
break;
case
```

```
ResponseEnum.MAIL_INBOX:
_gameController.handleMailInboxResponse(MailInboxResponse(response));
break;
case ResponseEnum.MAIL_DETAIL:
_gameController.handleMailDetailResponse(MailDetailResponse(response));
break;
}
break;
case ProtocolEnum.CHAT_CLIENT:
switch (response.header)
{
case ResponseEnum.CHAT_RESPONSE:
_chatController.recievedMessage(ChatResponse(response));
break;
case ResponseEnum.CHAT_BASELINE:
_gameController.handleChatBaselineResponse(ChatBaselineResponse(response));
break;
case ResponseEnum.CHAT_EVENT:
//Do your own stuff here, Louis.
break;
}
break;
case ProtocolEnum.LEADERBOARD_CLIENT:
switch (response.header)
{
case ResponseEnum.LEADERBOARD:
_gameController.handleLeaderboardUpdate(LeaderboardResponse(response));
break;
case ResponseEnum.PLAYER_PROFILE:
_gameController.handlePlayerProfile(PlayerProfileResponse(response));
break;
case ResponseEnum.WARFRONT_UPDATE:
_gameController.handleWarfrontUpdate(WarfrontUpdateResponse(response));
break;
}
break;
case ProtocolEnum.ALLIANCE_CLIENT:
switch (response.header)
{
case ResponseEnum.ALLIANCE_BASELINE:
_gameController.handleAllianceBaselineResponse(AllianceBaselineResponse(response));
break;
case ResponseEnum.ALLIANCE_ROSTER:
_gameController.handleAllianceRosterResponse(AllianceRosterResponse(response));
break;
case ResponseEnum.PUBLIC_ALLIANCES_RESPONSE:
_gameController.handleAlliancePublicResponse(PublicAlliancesResponse(response));
break;
case ResponseEnum.GENERIC_ALLIANCE_RESPONSE:
_gameController.handleAllianceGenericResponse(AllianceGenericResponse(response));
break;
```

```
case ResponseEnum.ALLIANCE_INVITE:
_gameController.handleAllianceInviteResponse(AllianceInviteResponse(response));
}
break;
case ProtocolEnum.UNIVERSE_CLIENT:
switch (response.header)
{
case ResponseEnum.UNIVERSE_NEED_CHARACTER_CREATE:
//ExternalInterfaceAPI.logConsole("Imperium - a");

_gameController.handleUniverseNeedCharacterCreateResponse(UniverseNeedCharacterCreateRespons
break;
case ResponseEnum.UNIVERSE_SECTOR_LIST:
//ExternalInterfaceAPI.logConsole("Imperium - b");
_gameController.handleUniverseSectorListResponse(UniverseSectorListResponse(response));
break;
}
break;
}
if (response.isTicked)
{
//ExternalInterfaceAPI.logConsole("Imperium - response ticked");
var tickedResponse:ITickedResponse = ITickedResponse(response);
if (tickedResponse.isBaseline)
{
//we are listening to a new server
//clear out the old responses and ticks
_tickedResponses.length = 0;
_ticks.length = 0;
_time = 0;
_ticks.push(tickedResponse.tick, tickedResponse.timeStep);
//server tick to match the new server
SERVER_TICK = tickedResponse.tick;
updateSimulationTick();
INTERPOLATION = .5; //slow things down for the first frame to give the server a chance to send
the next update
handleTickedResponse(tickedResponse);
} else
{
if (tickedResponse.addTick)
addTick(tickedResponse.tick, tickedResponse.timeStep);
if (tickedResponse.tick <= SIMULATED_TICK)
handleTickedResponse(tickedResponse);
else
_tickedResponses.push(tickedResponse);
}
}
}

private
```

```
function handleTickedResponse( response:ITickedResponse ):void
{
switch (response.protocolID)
{
case ProtocolEnum.SECTOR_CLIENT:
switch (response.header)
{
case ResponseEnum.SECTOR_BASELINE:
_gameController.handleSectorBaselineResponse(SectorBaselineResponse(response));
break;
case ResponseEnum.SECTOR_ALWAYS_VISIBLE_UPDATE:

_gameController.handleSectorAlwaysVisibleUpdateResponse(SectorAlwaysVisibleUpdateResponse(resp
break;
case ResponseEnum.SECTOR_UPDATE:
_gameController.handleSectorUpdateResponse(SectorUpdateResponse(response));
break;
}
break;
case ProtocolEnum.BATTLE_CLIENT:
switch (response.header)
{
case ResponseEnum.BATTLE_BASELINE:
case ResponseEnum.BATTLE_UPDATE:
_gameController.handleBattleDataResponse(BattleDataResponse(response));
break;
case ResponseEnum.BATTLE_DEBUG_LINES:
_gameController.handleBattleDebugLinesResponse(BattleDebugLinesResponse(response));
break;
case ResponseEnum.BATTLE_HAS_ENDED:
_gameController.handleBattleEnded(BattleHasEndedResponse(response));
break;
}
break;
}
}

public function getRequest( protocolID:int, header:int ):IRequest
{
return PacketFactory.getRequest(protocolID, header);
}

public function send( request:IRequest ):void
{
if (_proxy)
{
_proxy.send(request);
_keepAliveTimer.reset();
_keepAliveTimer.start();
}
}
```

```
public function updateSimulationTime( time:Number ):Number
{
if (SIMULATED_TICK > SERVER_TICK)
return 0;
_tempInterp = time * INTERPOLATION;
if (_tempInterp > .1)
_tempInterp = .1;
_time += _tempInterp;
if (_time >= .1)
{
_time = _time - .1;
updateSimulationTick();
}
return _tempInterp;
}

public function disconnect():void
{
_tickedResponses = new Vector.<ITickedResponse>;
_ticks = [];
_time = 0;

_proxy.destroy();
_proxy = null;

_keepAliveTimer.reset();
_keepAliveTimer.stop();
}

private function addTick( tick:int, tickTime:int ):void
{
if (tick <= SERVER_TICK)
return;
/*trace(tick, tickTime, SIMULATED_TICK, getTimer() - _t);
_t = getTimer();*/
_ticks.push(tick, tickTime);
if (SIMULATED_TICK > SERVER_TICK)
{
SERVER_TICK = tick;
updateSimulationTick();
} else
SERVER_TICK = tick;
}

private function updateSimulationTick():void
{
if (_ticks.length > 0)
{
if
```

```
(SIMULATED_TICK >= SERVER_TICK)
_time = 0;
SIMULATED_TICK = _ticks.shift();
TIME_STEP = _ticks.shift();
_temp = SERVER_TICK - SIMULATED_TICK;
if (_temp <= 1)
_temp = 0;
_temp = _temp * .09; //Math.log(_temp) * .25;
INTERPOLATION = _replayDecoder ? 1.0 : ((TARGET_TIME_STEP / TIME_STEP) + _temp);
//trace(_temp, SERVER_TICK, SIMULATED_TICK, TARGET_TIME_STEP / TIME_STEP,
INTERPOLATION, TIME_STEP);

//check stored tick responses
var response:ITickedResponse;
while (_tickedResponses.length > 0 && _tickedResponses[0].tick <= SIMULATED_TICK)
{
response = _tickedResponses.shift();
if (response.protocolID == _proxy.protocolListener)
handleTickedResponse(response);
}
} else
SIMULATED_TICK++;
}

private function onProxyConnect( state:int ):void
{
if (state == TinCan.CONNECTED)
{
//ExternalInterfaceAPI.logConsole("Imperium IP proxy connected");
TimeLog.endTimeLog(TimeLogEnum.CONNECT_TO_PROXY);
var serverEvent:ServerEvent = new ServerEvent(ServerEvent.LOGIN_TO_ACCOUNT);
_eventDispatcher.dispatchEvent(serverEvent);
_keepAliveTimer.start();
} else if (state == TinCan.CONNECTION_LOST || state == TinCan.CONNECTION_FAILED)
{
//ExternalInterfaceAPI.logConsole("Imperium IP proxy connection lost...");
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
var nDisconnectView:DisconnectedView =
DisconnectedView(_viewFactory.createView(DisconnectedView));
nDisconnectView.titleText = 'CONNECTION LOST';
nDisconnectView.messageText = 'Your connection was no match for the Imperium!\nPlease
refresh your browser.\n\nError Message: 3025 Proxy Error';
viewEvent.targetView = nDisconnectView;
_eventDispatcher.dispatchEvent(viewEvent);
}
}

public function requestAllScores():void
{
_gameController.requestAllScores();
}
```

```
public function requestReplay( battleId:String ):void
{
if( Application.BATTLE_WEB_PATH == null )
{
return;
}
var absoluteUrl:String = Application.BATTLE_WEB_PATH + "?id="+battleId+".battle";
trace( "request", absoluteUrl );
AssetModel.instance.getFromCache( absoluteUrl, onBattleReplayData, LoadPriority.HIGH, true
);
}

private function onBattleReplayData( data:ByteArray ):void
{
trace( "received battle replay data" );
if( data.position==0)
{
// only uncompress once, the original data is changed
// after the first replay, data position will be at the end of the stream
data.uncompress();
}
data.position = 0;

_replayDecoder = new EightTrack();
_replayDecoder.init( data );

_replayDecoder.addResponseListener(handleResponse);
_replayDecoder.serverController = this;
_replayDecoder.onReceive(null); // TODO: this will immediately dispatch and parse all the data
}

public function cleanupBattle():void
{
if( _replayDecoder )
{
_replayDecoder.destroy();
_replayDecoder = null;
}
else
{
var battleDisconnect:ProxyConnectToBattleRequest =
ProxyConnectToBattleRequest(getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_CONNECT_TO_BATTLE));
send(battleDisconnect);
}
}

public function set lockRead( v:Boolean ):void { _proxy.lockRead = v; }

[Inject]
```

```
public function set chatController( v:ChatController ):void { _chatController = v; }

[Inject]
public function set eventDispatcher( v:IEventDispatcher ):void { _eventDispatcher = v; }

[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }

[Inject]
public function set settingsController( v:SettingsController ):void { _settingsController = v; }

[Inject]
public function set viewFactory( v:IViewFactory ):void { _viewFactory = v }

public function set protocolListener( protocolID:int ):void
{
if (_proxy.protocolListener != protocolID)
_tickedResponses.length = 0;
_proxy.protocolListener = protocolID;
}
}
}
```

File 8: igw\com\controller\SettingsController.as

```
package com.controller
{
import com.Application;
import com.controller.sound.SoundController;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.model.chat.ChatModel;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.service.server.outgoing.starbase.StarbaseSetClientSettingsRequest;
import com.ui.core.component.label.Label;

import flash.display.StageDisplayState;
import flash.events.TimerEvent;
import flash.utils.Timer;

import org.console.Cc;
import org.starling.core.Starling;

public class SettingsController
{
private var _agentGreetingsViewed:int;
private var _chatModel:ChatModel;
private var _fleetModel:FleetModel;
private
```

```
var _sendTimer:Timer;
private var _serverController:ServerController;
private var _soundController:SoundController;

[PostConstruct]
public function init():void
{
_sendTimer = new Timer(1000)
_sendTimer.addEventListener(TimerEvent.TIMER, onTimerFinished, false, 0, true);
Cc.addSlashCommand('fullscreen', toggleFullScreen);
Cc.addSlashCommand('loc', onLoc);
}

public function toggleFullScreen():void
{
if (Application.STAGE.displayState == StageDisplayState.NORMAL)
Application.STAGE.displayState = StageDisplayState.FULL_SCREEN_INTERACTIVE;
else
Application.STAGE.displayState = StageDisplayState.NORMAL;

startTimer();
}

public function toggleSFXMute():void
{
_soundController.toggleSFXMute();
startTimer();
}

public function toggleMusicMute():void
{
_soundController.toggleMusicMute();
startTimer();
}

public function setSFXVolume( v:Number ):void
{
_soundController.setSFXVolume(v);
startTimer();
}

public function setMusicVolume( v:Number ):void
{
_soundController.setMusicVolume(v);
startTimer();
}

public function save( stuff:* = null ):void { startTimer(); }

private function startTimer():void
{
```

```
if (_sendTimer.running)
_sendTimer.reset();

_sendTimer.start();
}

private function onTimerFinished( e:TimerEvent ):void
{
_sendTimer.stop();
sendSettings();
}

private function writeFleetGroupsToSettings():Object
{
var fleetGroups:Object = {};

for each (var fleetVO:FleetVO in _fleetModel.fleets)
{
if (fleetVO.fleetGroupData.length == 0)
continue;

fleetGroups[fleetVO.id] = {};

for (var groupKey:String in fleetVO.fleetGroupData)
{
var groupIdxs:String = fleetVO.fleetGroupData[groupKey];
if (groupIdxs && groupIdxs.length > 0)
fleetGroups[fleetVO.id][groupKey] = groupIdxs;
}
}

return fleetGroups;
}

private function readFleetGroupsFromSettings( settings:Object ):void
{
if (!settings.hasOwnProperty("fleetGroups"))
return;
var fleetGroups:Object = settings["fleetGroups"];
for each (var fleetVO:FleetVO in _fleetModel.fleets)
{
if (fleetGroups.hasOwnProperty(fleetVO.id))
{
for (var groupKey:String in fleetGroups[fleetVO.id])
{
var groupIdxsString:String = fleetGroups[fleetVO.id][groupKey];
if (groupIdxsString && groupIdxsString.length > 0)
{
if (groupIdxsString.indexOf(',') != -1)
fleetVO.fleetGroupData[groupKey]
```

```
= groupIdxsString.split(',').join('');
else
fleetVO.fleetGroupData[groupKey] = groupIdxsString;
}
}
}
}
}

private function sendSettings():void
{
var areSFXMuted:Boolean = _soundController.areSFXMuted;
var sfxVolume:Number = _soundController.sfxVolume;
var isMusicMuted:Boolean = _soundController.isMusicMuted;
var musicVolume:Number = _soundController.musicVolume;
var stageState:String = Application.STAGE.displayState;
var defaultChatChannel:int = -1;

if (_chatModel.defaultChannel != null)
defaultChatChannel = _chatModel.defaultChannel.channelID;

var agentGreetingsViewed:int = _agentGreetingsViewed;

var settings:Object =
{
'areSFXMuted':areSFXMuted,
'sfxVolume':sfxVolume,
'isMusicMuted':isMusicMuted,
'musicVolume':musicVolume,
'stageState':stageState,
'defaultChatChannel':defaultChatChannel,
'agentGreetingsViewed':agentGreetingsViewed,
'fleetGroups':writeFleetGroupsToSettings()
};

var sendSettings:StarbaseSetClientSettingsRequest =
StarbaseSetClientSettingsRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_SET_CLIENT_SETTINGS));
sendSettings.settings = JSON.stringify(settings);
_serverController.send(sendSettings);
}

public function setSettings( settings:Object ):void
{
if (settings)
{
var areSFXMuted:Boolean = Boolean(settings['areSFXMuted']);
var sfxVolume:Number = Number(settings['sfxVolume']);
var isMusicMuted:Boolean = Boolean(settings['isMusicMuted']);
var musicVolume:Number = Number(settings['musicVolume']);

var
```

```
stageState:String = String(settings['stageState']);
//var defaultChatChannel:int = int(settings['defaultChatChannel']);
var agentGreetingsViewed:int = int(settings['agentGreetingsViewed']);

readFleetGroupsFromSettings(settings);

if (areSFXMuted != _soundController.areSFXMuted)
_soundController.toggleSFXMute();

if (!isNaN(sfxVolume))
{
if (sfxVolume != _soundController.sfxVolume)
_soundController.setSFXVolume(sfxVolume);
} else
_soundController.setSFXVolume(0.5)


if (isMusicMuted != _soundController.isMusicMuted)
_soundController.toggleMusicMute();

if (!isNaN(musicVolume))
{
if (musicVolume != _soundController.musicVolume)
{
_soundController.setMusicVolume(musicVolume);
}
} else
_soundController.setMusicVolume(0.5);

//_chatModel.overrideDefaultChannelByChannelID(defaultChatChannel);
_agentGreetingsViewed = agentGreetingsViewed;
} else
_agentGreetingsViewed = 0;

_chatModel.onDefaultChannelUpdated.add(save);
}

public function hasAgentGreetingBeenViewed( agentID:int ):Boolean
{
var bit:int = _agentGreetingsViewed & (1 << agentID);

if (bit != 0)
return true;
else
return false;
}

public function setAgentGreetingViewed( agentID:int ):void
{
_agentGreetingsViewed |= 1 << agentID;
save();
```

```
}

private function onLoc( cmd:String ):void
{
if (cmd == 'on')
Label.showUnlocalizedString = true;
else if (cmd == 'off')
Label.showUnlocalizedString = false;
}

public function giveServerController( v:ServerController ):void { _serverController = v; }

public function get agentGreetingsViewed():int { return _agentGreetingsViewed; }

[Inject]
public function set chatModel( v:ChatModel ):void { _chatModel = v; }

[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }

[Inject]
public function set soundController( v:SoundController ):void { _soundController = v; }
}
}



-----------------------------------
File 9: igw\com\controller\command\BattleAlertCommand.as
package com.controller.command
{
import com.controller.ServerController;
import com.event.BattleEvent;
import com.presenter.battle.IBattlePresenter;
import com.service.server.incoming.battle.BattleDataResponse;
import com.service.server.incoming.battle.BattleHasEndedResponse;
import com.service.server.incoming.data.BattleData;
import com.ui.modal.battle.BattleEndView;
import com.ui.modal.battle.BattleStartView;

import org.parade.core.IViewFactory;
import org.parade.core.ViewController;
import org.parade.core.ViewEvent;
import org.robotlegs.extensions.presenter.impl.Command;

public class BattleAlertCommand extends Command
{
[Inject]
public var event:BattleEvent;
[Inject]
```

```
public var viewController:ViewController;
[Inject]
public var viewFactory:IViewFactory;
[Inject]
public var presenter:IBattlePresenter;

override public function execute():void
{
var viewEvent:ViewEvent
switch (event.type)
{
case BattleEvent.BATTLE_COUNTDOWN:
var battleStartResponse:BattleData = BattleData(event.response);
var battleStart:BattleStartView;
if (viewController.getView(BattleStartView))
{
battleStart = BattleStartView(viewController.getView(BattleStartView));
battleStart.update(ServerController.SIMULATED_TICK, battleStartResponse.battleStartTick);
} else
{
battleStart = BattleStartView(viewFactory.createView(BattleStartView));
battleStart.update(ServerController.SIMULATED_TICK, battleStartResponse.battleStartTick);
viewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetView = battleStart;
dispatch(viewEvent);
}
break;
case BattleEvent.BATTLE_STARTED:
presenter.onBattleStarted();
break;
case BattleEvent.BATTLE_ENDED:
presenter.onBattleEnded();
var response:BattleHasEndedResponse = BattleHasEndedResponse(event.response);
var battleEnd:BattleEndView = BattleEndView(viewFactory.createView(BattleEndView));
battleEnd.battleID = response.battleKey;
battleEnd.victors = response.victors;
battleEnd.lootedAlloyAmount = response.alloyLoot;
battleEnd.lootedCreditsAmount = response.creditBounty;
battleEnd.lootedEnergyAmount = response.energyLoot;
battleEnd.lootedSyntheticsAmount = response.syntheticLoot;
battleEnd.blueprintProtoName = response.blueprintReward;
battleEnd.cargoFull = response.cargoFull;
viewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetView = battleEnd;
dispatch(viewEvent);
break;
}
}
}
}
```

File 10: igw\com\controller\command\ContextLostCommand.as

```
package com.controller.command
{
import com.Application;
import com.enum.TypeEnum;
import com.event.StateEvent;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.fsm.FSM;
import com.game.entity.components.shared.fsm.Forcefield;
import com.game.entity.systems.interact.BattleInteractSystem;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.shared.background.BackgroundSystem;
import com.model.asset.AssetModel;
import com.util.RangeBuilder;
import com.util.RouteLineBuilder;

import flash.display.BitmapData;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;
import org.starling.text.BitmapFont;
import org.starling.text.TextField;
import org.starling.textures.Texture;

public class ContextLostCommand extends Command
{
[Inject]
public var assetModel:AssetModel;
[Inject]
public var rangeBuilder:RangeBuilder;
[Inject]
public var routeBuilder:RouteLineBuilder;
[Inject]
public var game:Game;

/**
* Stage3D context was lost and all textures were lost with it.
* We need to reload and rebuild everything to account for this
*/
override public function execute():void
{
//set up the OpenSansBold bitmap font
var ac:Class = Class(getDefinitionByName('OpenSansBoldSpriteSheet'));
var bmd:BitmapData = BitmapData(new ac());
ac
```

```
= Class(getDefinitionByName('FontsMain'))['OpenSansBoldXML'];
var xml:XML = XML(new ac());
var bf:BitmapFont = new BitmapFont(Texture.fromBitmapData(bmd, false), xml);
TextField.registerBitmapFont(bf, 'OpenSansBoldBitmap');

//set the ready flag of all entity animations to false so that spritesheets will be reloaded
var entities:Dictionary = game.allEntities;
for each (var entity:Entity in entities)
{
if (entity.has(Animation))
Animation(entity.get(Animation)).deviceLostContext();
if (entity.has(FSM))
{
var fsm:FSM = FSM(entity.get(FSM));
if (fsm.component is Forcefield)
Forcefield(fsm.component).state = Forcefield.POWER_OFF;
}
}

//remove all the current spritepacks which will force them to be recreated
assetModel.removeAllSpritePacks();

//force the starfield to be recreated
var _backgroundSystem:BackgroundSystem =
BackgroundSystem(game.getSystem(BackgroundSystem));
if (_backgroundSystem)
_backgroundSystem.uninitialize();

//clear and rebuild ship and building ranges
rangeBuilder.cleanup();
if (Application.STATE == StateEvent.GAME_BATTLE || Application.STATE ==
StateEvent.GAME_BATTLE_INIT)
BattleInteractSystem(game.getSystem(BattleInteractSystem)).buildRanges();
else if (Application.STATE == StateEvent.GAME_STARBASE)
StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem)).updateRanges();

//clear the data for the routeline so that it gets rebuilt
assetModel.removeGameAssetData(TypeEnum.ROUTE_LINE);
if (Application.STATE == StateEvent.GAME_SECTOR)
routeBuilder.drawRouteLine();
}

}
}

-----------------------------------
File 11: igw\com\controller\command\FTECommand.as
package com.controller.command
{
import com.Application;
import
```

```
com.controller.GameController;
import com.controller.ServerController;
import com.controller.fte.FTEController;
import com.controller.sound.SoundController;
import com.controller.toast.ToastController;
import com.controller.transaction.TransactionController;
import com.enum.AudioEnum;
import com.enum.CategoryEnum;
import com.enum.ToastEnum;
import com.enum.TypeEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.FTEEvent;
import com.event.StateEvent;
import com.event.ToastEvent;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.nodes.shared.grid.GridNode;
import com.game.entity.systems.interact.BattleInteractSystem;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.shared.grid.GridSystem;
import com.model.asset.AssetModel;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.mission.MissionInfoVO;
import com.model.mission.MissionModel;
import com.model.mission.MissionVO;
import com.model.scene.SceneModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.presenter.shared.IUIPresenter;
import com.service.ExternalInterfaceAPI;
import com.service.server.outgoing.battle.BattlePauseRequest;
import com.ui.alert.FTEOverlayView;
import com.ui.core.component.contextmenu.ContextMenu;
import com.ui.hud.shared.PlayerView;
import com.ui.modal.battle.BattleEndView;
import com.ui.modal.intro.FTETipView;
import com.ui.modal.intro.PulseScaleEffect;
import com.ui.modal.mission.FTEDialogueView;
import com.ui.modal.shipyard.ComponentSelection;
import com.ui.modal.shipyard.ShipyardView;

import flash.display.DisplayObject;
import flash.filters.BitmapFilterQuality;
import flash.filters.GlowFilter;
import flash.geom.Point;
import flash.geom.Rectangle;
import
```

```
flash.utils.getDefinitionByName;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.Node;
import org.ash.core.NodeList;
import org.parade.core.IView;
import org.parade.core.IViewFactory;
import org.parade.core.ViewController;
import org.parade.core.ViewEvent;
import org.parade.util.DeviceMetrics;
import org.robotlegs.extensions.presenter.impl.Command;
import org.shared.ObjectPool;

public class FTECommand extends Command
{
private static const GLOW_FILTER:GlowFilter = new GlowFilter(0xffff00, 0.8, 5, 5, 4,
BitmapFilterQuality.HIGH);
private static const NEUTRAL:int = 0;
private static const SERIOUS:int = 1;
private static const SMILE:int = 2;
private static const SURPRISED:int = 3;

private static var _glowEntity:Entity;
private static var _glowUI:DisplayObject;
private static var _scaleUI:PulseScaleEffect;

[Inject]
public var assetModel:AssetModel;
[Inject]
public var event:FTEEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var fteController:FTEController;
[Inject]
public var game:Game;
[Inject]
public var gameController:GameController;
[Inject]
public var missionModel:MissionModel;
[Inject]
public var sceneModel:SceneModel;
[Inject]
public var serverController:ServerController;
[Inject]
public var soundController:SoundController;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public
```

```
var toastController:ToastController;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var uiPresenter:IUIPresenter;
[Inject]
public var viewController:ViewController;
[Inject]
public var viewFactory:IViewFactory;

private var _dialogueView:FTEDialogueView;
private var _overlayView:FTEOverlayView;

override public function execute():void
{
//if we added a glow to an entity last step, remove it
_glowEntity && removeGlowFromEntity(_glowEntity);
removeGlowFromUI();
removeScaleFromUI();

//determine which type of fte event this is and act accordingly
switch (event.type)
{
case FTEEvent.FTE_COMPLETE:
ExternalInterfaceAPI.popPixel(2)
showHideOverlay(false);
showHideDialog(false);
break;
case FTEEvent.FTE_STEP:
showHideOverlay(true);
showHideDialog(true);
addCodeTrigger();
break;
}
}

private function showHideOverlay( show:Boolean = true ):void
{
_overlayView = FTEOverlayView(viewController.getView(FTEOverlayView));
if (_overlayView)
{
if (_overlayView.presenter != null)
_overlayView.destroy();
else
viewController.removeFromQueue(FTEOverlayView);
}

if (show)
{
var view:IView;
var
```

```
viewAlreadyExisted:Boolean = true;
if (event.step.viewClass != null)
{
view = viewController.getView(event.step.viewClass);
if (!view)
{
viewAlreadyExisted = false;
view = viewFactory.createView(event.step.viewClass);
viewFactory.notify(view);
}
}

_overlayView = FTEOverlayView(viewFactory.createView(FTEOverlayView));
_overlayView.clickToContinue = event.step.cutout == null;
_overlayView.fteStepVO = event.step;
_overlayView.view = view;
_overlayView.viewAlreadyExisted = viewAlreadyExisted;
viewFactory.notify(_overlayView);
}
}

private function showHideDialog( show:Boolean = true ):void
{
//show / hide the dialogue view. see if the view already exists before making a new one
_dialogueView = FTEDialogueView(viewController.getView(FTEDialogueView));
if (!show && _dialogueView)
_dialogueView.destroy();
else if (_dialogueView || (event.step.dialog != null && event.step.dialog != ""))
{
var dialogueAlreadyExisted:Boolean = true;
if (!_dialogueView)
{
dialogueAlreadyExisted = false;
_dialogueView = FTEDialogueView(viewFactory.createView(FTEDialogueView));
} else if (_dialogueView.parent)
_dialogueView.parent.addChild(_dialogueView);
var info:MissionInfoVO = ObjectPool.get(MissionInfoVO);
if (event.step.dialog != null && event.step.dialog != "")
{
info.addDialog(event.step.dialog);
info.addTitle(event.step.titleText.toLocaleUpperCase(), 0xffb128);
}

switch (event.step.mood)
{
case NEUTRAL:
info.addImages("AITS_new.png", "AITS_new.png", "AITS_new.png");
break;
case SERIOUS:
info.addImages("AITS_Flirty.png", "AITS_Flirty.png", "AITS_Flirty.png");
break;
```

```
case SMILE:
info.addImages("AITS_Smile.png", "AITS_Smile.png", "AITS_Smile.png");
break;
case SURPRISED:
info.addImages("AITS_Surprised.png", "AITS_Surprised.png", "AITS_Surprised.png");
break;
}
info.currentProgress = event.step.currentStep;
info.progressRequired = event.step.totalSteps;
_dialogueView.nextButtonEnabled = event.step.cutout == null;
_dialogueView.info = info;
_dialogueView.hideViews();
_dialogueView.unhideViews();
if (event.step.voiceOver && event.step.voiceOver != "")
soundController.playSound(event.step.voiceOver, .47);

if (!dialogueAlreadyExisted)
viewFactory.notify(_dialogueView);

if (_overlayView)
_overlayView.dialogueView = _dialogueView;
}
}

private function addCodeTrigger():void
{
//hide or show the leftside bridge
var view:*;

if (!event.step.trigger)
return;
var entity:Entity;
var mission:MissionVO;
var p:Point, p2:Point, p3:Point;
var params:Array;
var position:Position;
var pauseRequest:BattlePauseRequest;
var triggers:Array = event.step.trigger.split(',');
var haltCommands:Boolean;
var anim:Animation;

for (var a:int = 0; a < triggers.length; a++)
{
if (haltCommands)
break;

params = triggers[a].split('|');
switch (params[0])
{
//
```

```
Moves the view to the center point between the player and enemy fleets
case "centerOnFleets":
entity = findFleet(true);
p = (entity.get(Position) as Position).position;
entity = findFleet(false);
p2 = (entity.get(Position) as Position).position;
p3 = Point.interpolate(p, p2, 0.5);
BattleInteractSystem(game.getSystem(BattleInteractSystem)).moveToLocation(p3.x, p3.y, .5);
break;

case "centerOnTransgate":
entity = findTransgate();

if (entity)
{
position = entity.get(Position);
SectorInteractSystem(game.getSystem(SectorInteractSystem)).moveToLocation(position.x,
position.y, 0.5);

p = new Point(DeviceMetrics.WIDTH_PIXELS * .5, DeviceMetrics.HEIGHT_PIXELS * .5);
anim = entity.get(Animation);
if (anim)
{
p.x += anim.offsetX * 0.5;
}

event.step.arrowPosition = p;
event.step.arrowRotation = 180;
if (_overlayView && _overlayView.parent)
_overlayView.showArrow();
}
break;

//checks to see if the fleet is in battle, if not it notifies gamecontroller that the fte is waiting on it
case "checkFleetInBattle":
var inBattle:Boolean = false;
var fleets:Vector.<FleetVO> = fleetModel.fleets;
for (var i:int = 0; i < fleets.length; i++)
{
if (fleets[i].inBattle)
{
inBattle = true;
break;
}
}
if (!inBattle)
{
//fleet is not yet in battle. let the gamecontroller know
//when the fleet enters battle the gamecontroller will progress the fte
if (_dialogueView)
_dialogueView.nextButtonEnabled
```

```
= false;
if (_overlayView)
_overlayView.clickToContinue = false;
gameController.inFTE = true;
}
break;

//check to see if the player is on the mission needed for the next step
case "checkForMission":
mission = missionModel.currentMission;
var names:Array = event.step.missionName.split(',');
var index:int = names.indexOf(mission.name);
if (index > -1)
{
fteController.nextStep();
}
break;

//closes a view specified in the parameters
case "closeView":
var viewClass:Class = Class(getDefinitionByName(params[1]));
view = viewController.getView(viewClass);
if (view)
view.destroy();
break;

//disables the next button in the fte
case "disableNext":
if (_dialogueView)
_dialogueView.nextButtonEnabled = false;
if (_overlayView)
_overlayView.clickToContinue = false;
break;

case "disableHUD":
uiPresenter.hudEnabled = false;
break;

case "enableHUD":
uiPresenter.hudEnabled = true;
break;

//a few missions need the client to progress them. use this trigger to do so
case "forceMissionComplete":
mission = missionModel.currentMission;
if (mission)
{
transactionController.missionStepRequest(mission.name, 1);
}
break;

//call
```

```
this to have the game follow a fleet as it moves on the screen
case "followFleet":
entity = findFleet(true);
if (Application.STATE == StateEvent.GAME_BATTLE)
{
BattleInteractSystem(game.getSystem(BattleInteractSystem)).inFTE = true;
BattleInteractSystem(game.getSystem(BattleInteractSystem)).followEntity = entity;
} else
{
SectorInteractSystem(game.getSystem(SectorInteractSystem)).inFTE = true;
SectorInteractSystem(game.getSystem(SectorInteractSystem)).followEntity = entity;
}
break;

//sometimes an fte step needs to be forced onto the next step without any user interaction
case "forceNextStep":
fteController.nextStep();
break;

case "hideOverlay":
showHideOverlay(false);
break;
case "hideDialogue":
showHideDialog(false);
break;

case "highlightUI":
highlightUI(params[1]);
break;

//call this to have the game follow a fleet as it moves on the screen
case "ignoreStoreOffset":
if (_overlayView)
_overlayView.ignoreStoreOffset = true;
break;

//closes a view specified in the parameters
case "killToast":
toastController.killCurrentToast();
break;

case "moveToEnemyFleet":
entity = findFleet(false);
position = entity.get(Position);
BattleInteractSystem(game.getSystem(BattleInteractSystem)).moveToLocation(position.x,
position.y, .5);
break;
case "moveToFleet":
if (Application.STATE == StateEvent.GAME_BATTLE)
{
entity
```

```
= findFleet(true);
position = entity.get(Position);
BattleInteractSystem(game.getSystem(BattleInteractSystem)).moveToLocation(position.x,
position.y, .5);
} else
{
entity = findFleet(true);
if (entity)
{
SectorInteractSystem(game.getSystem(SectorInteractSystem)).selectEntity(entity);
} else
{
fleets = fleetModel.fleets;
for (i = 0; i < fleets.length; i++)
{
if (fleets[i].sector != '')
{

SectorInteractSystem(game.getSystem(SectorInteractSystem)).jumpToLocation(fleets[i].sectorLocationX,
fleets[i].sectorLocationY);
break;
}
}
}
}
break;
case "notifyOnBattleEnd":
gameController.inFTE = true;
break;
case "pauseBattle":
pauseRequest =
BattlePauseRequest(serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_PAUSE));
pauseRequest.pause = true;
serverController.send(pauseRequest);
break;

//let StarbaseInteractSystem know that the fte is running and the player is placing a building
case "placeBuilding":
StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem)).inFTE = true;
//destroy the overlay so the player can click
showHideOverlay(false);
break;

case "playThemeMusic":
soundController.playSound(AudioEnum.AFX_BG_MAIN_THEME, 0.07, 0, 100);
break;

//points to a specific building. pass the itemClass in as a parameter ie.
pointToBuilding|CommandCenter
case
```

```
"pointToBuilding":
entity = findBuilding(params[1]);
if (entity)
{
position = entity.get(Position);

StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem)).moveToLocation(position.x,
position.y, .5);
p = new Point(DeviceMetrics.WIDTH_PIXELS * .5, DeviceMetrics.HEIGHT_PIXELS * .5);
p2 = p.clone();
anim = entity.get(Animation);
if (anim)
{
p2.x += anim.offsetX * 0.5;
}

event.step.arrowPosition = p2;
event.step.arrowRotation = 180;
if (_overlayView && _overlayView.parent)
_overlayView.showArrow();
}
break;

case "pointToCenterOfScreen":
pointToCenterOfScreen();
break;

case "pressWASD":
BattleInteractSystem(game.getSystem(BattleInteractSystem)).inFTE = true;
break;

//when Application.STATE changes the fte will progress to the next step
case "progressOnStateChange":
fteController.progressStepOnStateChange = true;
break;

case "rolloverTrigger":
if (_overlayView)
_overlayView.rolloverTrigger();
break;

//called to select the first item in a context menu. TODO: Add the index of the item to select in
the parameters
case "selectContextMenu":
var contextMenu:ContextMenu = ContextMenu(viewController.getView(ContextMenu));
if (contextMenu)
{
contextMenu.destroyOnRollout = false;
fteController.closeContext = contextMenu;
}
break;
```

```
//selects the players starbase
case "selectBase":
selectEntity(findBase(), 100, 100, 0);
SectorInteractSystem(game.getSystem(SectorInteractSystem)).inFTE = true;
break;

//selects a specific building. pass the itemClass in as a parameter ie.
selectBuilding|CommandCenter
case "selectBuilding":
entity = findBuilding(params[1]);
if (entity)
{
position = entity.get(Position);

StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem)).moveToLocation(position.x,
position.y, .5);
selectEntity(entity, 200, 200);
StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem)).inFTE = true;
}
if (_dialogueView)
_dialogueView.nextButtonEnabled = false;
if (_overlayView)
_overlayView.clickToContinue = false;
break;

//selects a fleet
case "selectEnemyFleet":
case "selectFleet":
selectEntity(findFleet(params[0] == "selectFleet"), 100, 100);
if (Application.STATE == StateEvent.GAME_BATTLE)
BattleInteractSystem(game.getSystem(BattleInteractSystem)).inFTE = true;
else
SectorInteractSystem(game.getSystem(SectorInteractSystem)).inFTE = true;
break;

//ship slots differ for each faction so we can't rely on the normal way we do cutouts and arrows.
//this trigger will find a slot of the specified type and highlight it
case "selectShipSlot":
var shipyard:ShipyardView = ShipyardView(viewController.getView(ShipyardView));
if (shipyard.components[0].parent.x == 0 && shipyard.components[0].parent.y == 0)
shipyard.addLoadCallback(onShipyardLoad);
else
onShipyardLoad();
break;

case "showTipModal":
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = FTETipView;
dispatch(viewEvent);
```

```
uiPresenter.playSound("sounds/vo/fte/FTE_001.mp3");
break;

case "toastImage":
var asset:String = "assets/" + params[1];
var text:String = params[2];
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.data = {url:asset, text:text};
toastEvent.toastType = ToastEnum.FTE_REWARD;
dispatch(toastEvent);
break;

case "unpauseBattle":
pauseRequest =
BattlePauseRequest(serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_PAUSE));
pauseRequest.pause = false;
serverController.send(pauseRequest);
break;

case "waitForMove":
BattleInteractSystem(game.getSystem(BattleInteractSystem)).toggleFTEProgressOnMove();
break;

default:
throw new Error("No such FTE command: " + triggers[a]);
}
}
}

private function findBase():Entity
{
var entity:Entity;
var nodes:NodeList;
var sectorInteractSystem:SectorInteractSystem =
SectorInteractSystem(game.getSystem(SectorInteractSystem));
nodes = sectorInteractSystem.owned;
for (var node:Node = nodes.head; node; node = node.next)
{
if (Detail(node.entity.get(Detail)).category == CategoryEnum.SECTOR)
{
entity = node.entity;
break;
}
}
return entity;
}

private
```

```
function findBuilding( type:String ):Entity
{
var buildings:Vector.<BuildingVO> = starbaseModel.buildings;
var entity:Entity;
for (var i:int = 0; i < buildings.length; i++)
{
if (buildings[i].itemClass == type)
{
entity = game.getEntity(buildings[i].id);
break;
}
}
return entity;
}

private function findFleet( owned:Boolean ):Entity
{
var entity:Entity;
var node:Node;
var nodes:NodeList;
if (Application.STATE == StateEvent.GAME_BATTLE)
{
var battleSystem:BattleInteractSystem =
BattleInteractSystem(game.getSystem(BattleInteractSystem));
nodes = owned ? battleSystem.owned : battleSystem.enemy;
for (node = nodes.head; node; node = node.next)
{
if (Detail(node.entity.get(Detail)).category == CategoryEnum.SHIP)
{
entity = node.entity;
break;
}
}
} else if (owned)
{
var sectorInteractSystem:SectorInteractSystem =
SectorInteractSystem(game.getSystem(SectorInteractSystem));
nodes = sectorInteractSystem.owned;
for (node = nodes.head; node; node = node.next)
{
if (Detail(node.entity.get(Detail)).category == CategoryEnum.SHIP)
{
entity = node.entity;
break;
}
}
}
return entity;
}

private
```

```
function findTransgate():Entity
{
var entity:Entity;

// This will only work in sector mode, because it's the only place transgates are found, naturally.
var gridSystem:GridSystem = GridSystem(game.getSystem(GridSystem));
for (var node:GridNode = gridSystem.nodes.head; node; node = node.next)
{
var detail:Detail = node.entity.get(Detail)

switch (detail.type)
{
case TypeEnum.TRANSGATE_IGA:
case TypeEnum.TRANSGATE_SOVEREIGNTY:
case TypeEnum.TRANSGATE_TYRANNAR:
entity = node.entity;
break;
}

if (entity)
break;
}

return entity;
}

private function onShipyardLoad():void
{
var shipyard:ShipyardView = ShipyardView(viewController.getView(ShipyardView));
var triggers:Array = event.step.trigger.split(',');
var params:Array = triggers[0].split('|');
var p:Point;
for each (var component:ComponentSelection in shipyard.components)
{
if (component.slotType == params[1])
{
// These coords are reversed because the view is rotated 90 degrees
p = new Point(component.parent.x - component.y - component.height, component.parent.y +
component.x);
var ap:Point = p.clone();
ap.y += component.width * .5;
event.step.arrowPosition = ap;
event.step.arrowRotation = 0;
event.step.cutout = new Rectangle(p.x, p.y, component.height, component.width);
if (_overlayView && _overlayView.parent)
{
_overlayView.showArrow();
if (params.length == 2)
_overlayView.showCutout();
else
{
```

```
addGlowToUI(component);
event.step.arrowPosition = null;
}
_overlayView.clickToContinue = params.length == 3;
_dialogueView.nextButtonEnabled = params.length == 3;
}
break;
}
}
shipyard.addLoadCallback(null);
}

private function pointToCenterOfScreen():void
{
var width:Number = DeviceMetrics.WIDTH_PIXELS * .5;
var height:Number = DeviceMetrics.HEIGHT_PIXELS * .5;
var p:Point = new Point(width, height);
event.step.arrowPosition = p;
event.step.arrowRotation = 90;

event.step.cutout = new Rectangle(p.x - width, p.y - height, width * 2, height * 2);
if (_overlayView && _overlayView.parent)
{
_overlayView.showArrow();
_overlayView.showCutout();
_overlayView.mouseChildren = _overlayView.mouseEnabled = false;
}
}

private function selectEntity( entity:Entity, width:int, height:int, yOffset:int = 0 ):void
{
if (entity)
{
var position:Position = entity.get(Position);
var p:Point = new Point(DeviceMetrics.WIDTH_PIXELS * .5, DeviceMetrics.HEIGHT_PIXELS *
.5);

var p2:Point = p.clone();
var anim:Animation = entity.get(Animation);
if (anim)
{
p2.x += anim.offsetX * 0.5;
}

p2.y += yOffset;

event.step.arrowPosition = p2;
event.step.arrowRotation = 180;

event.step.cutout
```

```
= new Rectangle(p.x - (width / 2), p.y - (height / 2), width, height);
if (_overlayView && _overlayView.parent)
{
_overlayView.showArrow();
_overlayView.showCutout();
}
addClickGlowToEntity(entity);
}
}

private function highlightUI( ui:String ):void
{
var uiObj:DisplayObject;
var view:IView;
switch (ui)
{
case "lootHolder":
view = viewController.getView(BattleEndView);
if (view && view is BattleEndView)
uiObj = (view as BattleEndView).lootHolder;
if (uiObj)
addGlowToUI(uiObj);
break;

case "HardCurrency":
view = viewController.getView(PlayerView);
if (view && view is PlayerView)
uiObj = (view as PlayerView).premiumBg;
if (uiObj)
addScaleToUI(uiObj);
break;
}
}

private function addGlowToUI( ui:DisplayObject ):void
{
removeGlowFromUI();
_glowUI = ui;
_glowUI.filters = [GLOW_FILTER];
}

private function removeGlowFromUI():void
{
if (_glowUI)
{
_glowUI.filters = [];
_glowUI = null;
}
}

private
```

```
function addScaleToUI( ui:DisplayObject ):void
{
removeScaleFromUI();
_scaleUI = new PulseScaleEffect(ui, 1, 1.7, .4);
}

private function removeScaleFromUI():void
{
if (_scaleUI)
{
_scaleUI.destroy();
_scaleUI = null;
}
}

private function addClickGlowToEntity( entity:Entity ):void
{
_glowEntity = entity;
var animation:Animation = entity.get(Animation);
if (animation && animation.render)
animation.render.addGlow(0xf9da54, 6, 8, 1);
}

private function addGlowToEntity( entity:Entity ):void
{
_glowEntity = entity;
var animation:Animation = entity.get(Animation);
if (animation && animation.render)
animation.render.addGlow(0xffff00, 20, 1, 1);
}

private function removeGlowFromEntity( entity:Entity ):void
{
_glowEntity = null;
var animation:Animation = entity.get(Animation);
if (animation && animation.render)
animation.render.removeGlow();
}
}
}
```

File 12: igw\com\controller\command\GuestCommand.as

```
package com.controller.command
{
import com.Application;
import com.controller.GameController;
import com.event.ServerEvent;
import com.service.ExternalInterfaceAPI;

import
```

```
org.parade.core.IViewFactory;
import org.robotlegs.extensions.presenter.impl.Command;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

import com.ui.modal.information.GuestRestrictionView;

public class GuestCommand extends Command
{
[Inject]
public var event:ServerEvent;
[Inject]
public var viewFactory:IViewFactory;
[Inject]
public var gameController:GameController;

private static const _logger:ILogger = getLogger('GuestCommand');

override public function execute():void
{
switch (event.type)
{
case ServerEvent.GUEST_RESTRICTION:
if (viewFactory)
{
var window:GuestRestrictionView =
GuestRestrictionView(viewFactory.createView(GuestRestrictionView));
viewFactory.notify(window);
}
break;
}
}
}
}

-----------------------------------
File 13: igw\com\controller\command\MissionCommand.as
package com.controller.command
{
import com.enum.ToastEnum;
import com.event.MissionEvent;
import com.event.ToastEvent;
import com.model.mission.MissionModel;
import com.presenter.starbase.IMissionPresenter;
import com.ui.modal.mission.DialogueView;

import org.parade.core.IViewFactory;
import org.robotlegs.extensions.presenter.impl.Command;

public
```

```
class MissionCommand extends Command
{
[Inject]
public var event:MissionEvent;
[Inject]
public var missionModel:MissionModel;
[Inject]
public var presenter:IMissionPresenter;
[Inject]
public var viewFactory:IViewFactory;

override public function execute():void
{
switch (event.type)
{
case MissionEvent.SHOW_REWARDS:
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.data = presenter.getMissionInfo(MissionEvent.MISSION_VICTORY);
toastEvent.toastType = ToastEnum.MISSION_REWARD;
dispatch(toastEvent);
presenter.acceptMissionReward();
break;
default:
var dialogueView:DialogueView = DialogueView(viewFactory.createView(DialogueView));
dialogueView.info = presenter.getMissionInfo(event.type);
dialogueView.state = event.type;
viewFactory.notify(dialogueView);
break;
}
}
}
}
```

File 14: igw\com\controller\command\PaywallCommand.as

```
package com.controller.command
{
import com.Application;
import com.controller.GameController;
import com.event.PaywallEvent;
import com.service.kongregate.KongregateAPI;
import com.service.facebook.FacebookAPI;
import com.ui.modal.paywall.PaywallView;
import com.service.ExternalInterfaceAPI;

import org.parade.core.IViewFactory;
import org.robotlegs.extensions.presenter.impl.Command;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public
```

```
class PaywallCommand extends Command
{
[Inject]
public var event:PaywallEvent;
[Inject]
public var viewFactory:IViewFactory;
[Inject]
public var gameController:GameController;
[Inject]
public var kongregateAPI:KongregateAPI;
[Inject]
public var facebookAPI:FacebookAPI;

private static const _logger:ILogger = getLogger('PaywallCommand');

override public function execute():void
{
switch (event.type)
{
case PaywallEvent.GET_PAYWALL:
_logger.debug('execute - PaywallEvent.GET_PAYWALL');
if (gameController)
gameController.requestPaywallPayouts();
break;
case PaywallEvent.OPEN_PAYWALL:
_logger.debug('execute - PaywallEvent.OPEN_PAYWALL');
if (viewFactory)
{
var paywall:PaywallView = PaywallView(viewFactory.createView(PaywallView));
viewFactory.notify(paywall);
paywall.setUp(event.paywallData);
}
break;
case PaywallEvent.BUY_ITEM:
_logger.debug('execute - PaywallEvent.BUY_ITEM');
if (gameController)
{
if (Application.NETWORK == Application.NETWORK_GOOGLEAPP)
gameController.requestPaymentVerification(event.externalTrkid, event.payoutId,
event.responseData, event.responseSignature);
else if (Application.NETWORK == Application.NETWORK_KONGREGATE)
gameController.requestPaymentVerification('', '', '', kongregateAPI.gameAuthToken);
else if (Application.NETWORK == Application.NETWORK_FACEBOOK)
gameController.requestPaymentVerification('', '', '', '');
}
break;
}
}
}
}
```

File 15: igw\com\controller\command\StarbaseCommand.as

```
package com.controller.command
{
import com.Application;
import com.controller.fte.FTEController;
import com.controller.transaction.TransactionController;
import com.event.BattleEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.model.fleet.FleetModel;
import com.model.motd.MotDModel;
import com.model.starbase.StarbaseModel;
import com.ui.alert.AttackAlert;

import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseCommand extends Command
{
[Inject]
public var event:StarbaseEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var fteController:FTEController;
[Inject]
public var motdModel:MotDModel;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var viewFactory:IViewFactory;

override public function execute():void
{
var battleEvent:BattleEvent;
var alert:AttackAlert;
var viewEvent:ViewEvent
switch (event.type)
{
case StarbaseEvent.ALERT_FLEET_BATTLE:
//inform the player that their fleet is under attack and ask if they want to defend it
alert = AttackAlert(viewFactory.createView(AttackAlert));
alert.battleServerAddress = event.battleServerAddress;
alert.fleetID = event.fleetID;
alert.fleetName = fleetModel.getFleet(event.fleetID).name;
viewFactory.notify(alert);
break;
```

```
case StarbaseEvent.ALERT_STARBASE_BATTLE:
if (Application.STATE == StateEvent.GAME_STARBASE && event.baseID ==
starbaseModel.currentBase.id)
{
//attack is happening on a base the player is currently viewing.
//player cannot make starbase changes while in battle so send straight in
battleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN, event.battleServerAddress);
dispatch(battleEvent);
} else
{
//popup up a notification asking the player if they want to defend their base
alert = AttackAlert(viewFactory.createView(AttackAlert));
alert.battleServerAddress = event.battleServerAddress;
viewFactory.notify(alert);
}
break;
case StarbaseEvent.ALERT_INSTANCED_MISSION_BATTLE:
battleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN, event.battleServerAddress);
dispatch(battleEvent);
break;
}
}
}
}
```

File 16: igw\com\controller\command\ToastCommand.as

```
package com.controller.command
{
import com.controller.fte.FTEController;
import com.controller.toast.ToastController;
import com.enum.CurrencyEnum;
import com.enum.ToastEnum;
import com.event.ToastEvent;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintVO;
import com.model.mission.MissionInfoVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.service.loading.LoadPriority;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.toast.ToastView;

import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;
import
```

```
org.robotlegs.extensions.presenter.impl.Command;
import org.shared.ObjectPool;

public class ToastCommand extends Command
{
//backgrounds
public static const BG_SMALL:String = "ToastSmallBMD";
public static const BG_LARGE:String = "ToastLargeBMD";
public static const BG_RESOURCES:String = "ToastResourcesBMD";

private var _palladiumAdded:String = 'CodeString.Shared.PalladiumAdded'; //Level
[[Number.Level]]
private var _level:String = 'CodeString.Shared.Level'; //Level [[Number.Level]]
private var _missionComplete:String = 'CodeString.Toast.MissionComplete'; //Mission Complete!
private var _congratulations:String = 'CodeString.Toast.Congratulations'; //Congratulations!
private var _reward:String = 'CodeString.Toast.Reward' //Reward:
private var _alert:String = 'CodeString.Toast.Alert'; //ALERT
private var _alliance:String = 'CodeString.Toast.AllianceTitle'; //Alliance Event
private var _blueprintTitle:String = 'CodeString.Toast.Blueprint'; //Blueprint Part Acquired
private var _achievementUnlocked:String = 'CodeString.Achievement.ToastTitle'; //Achievement
Unlocked!
private var _incomingMessage:String = 'CodeString.Toast.IncomingMessage'; //Incoming
Message
private var _newMission:String = 'CodeString.Toast.NewMission'; // New mission for you.

[Inject]
public var assetModel:AssetModel;
[Inject]
public var event:ToastEvent;
[Inject]
public var fteController:FTEController;
public var toast:ToastView;
[Inject]
public var toastController:ToastController;

override public function execute():void
{
if (fteController.running && !(event.toastType == ToastEnum.MISSION_REWARD ||
event.toastType == ToastEnum.FTE_REWARD))
return;

var assetVO:AssetVO;
var label:Label;
var title:Label;
toast = ObjectPool.get(ToastView);

switch (event.toastType)
{
case ToastEnum.FLEET_DOCKED:
{
toast.addBackground(BG_RESOURCES);
```

```
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0xfac973), 0, 10, event.nextString);
toast.addLabel("message", LabelFactory.createLabel(-1, 340, 23, 0xf0f0f0), 0, 50,
event.nextString);
toast.addLabel("alloy", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR, 105,
22, 0xf0f0f0), 225, 104,
StringUtil.commaFormatNumber(event.nextString), TextFormatAlign.LEFT);
toast.addLabel("credit", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR, 105,
22, 0xf0f0f0), 75, 104,
event.nextString, TextFormatAlign.LEFT);
toast.addLabel("energy", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR,
105, 22, 0xf0f0f0), 75, 139,
StringUtil.commaFormatNumber(event.nextString), TextFormatAlign.LEFT);
toast.addLabel("synthetic", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR,
105, 22, 0xf0f0f0), 225, 139,
StringUtil.commaFormatNumber(event.nextString), TextFormatAlign.LEFT);
break;
}

case ToastEnum.FTE_REWARD:
{
toast.addBackground(BG_LARGE);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0xfac973), 0, 10, event.data.text);
addImage("image", event.data.url, 0, 25, 340);
break;
}

case ToastEnum.LEVEL_UP:
{
toast.addBackground(BG_SMALL);
toast.addLabel("message", LabelFactory.createLabel(-1, 340, 22, 0xf0f0f0), 0, 20,
_congratulations);
label = LabelFactory.createLabel(-1, 340, 32, 0xfac973);
label.setTextWithTokens(_level, {'[[Number.Level]]':CurrentUser.level});
toast.addLabel("level", label, 0, 50);
break;
}

case ToastEnum.PALLADIUM_ADDED:
{
toast.addBackground(BG_LARGE);
var ty:Number = (toast.height - 121) / 2;
addImage("icon", "assets/Palladium_LG.png", 0, ty, 108, 121);
title = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DIALOG_TITLE, toast.width - 118,
-1, 0xf0f0f0)
toast.addLabel("message", title, 100, 20, _congratulations);
label = LabelFactory.createLabel(-1, toast.width - 118, 32, 0xfac973, true);
label.setTextWithTokens(_palladiumAdded,
{'[[Number.Amount]]':CurrentUser.vo.wallet.getPrevAddedAmount(CurrencyEnum.PREMIUM)});
toast.addLabel("level", label, 100, 50);
var
```

```
th:Number = title.textHeight + 10 + label.textHeight;
title.y = (toast.height - th) / 2;
label.y = title.y + title.textHeight + 10;

break;
}

case ToastEnum.MISSION_NEW:
{
toast.addBackground(BG_SMALL);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0xfac973), 0, 10,
_incomingMessage);
toast.addLabel("message", LabelFactory.createLabel(-1, 340, 23, 0xf0f0f0), 0, 50,
_newMission);
break;
}

case ToastEnum.MISSION_REWARD:
{
var missionInfo:MissionInfoVO = MissionInfoVO(event.data);
toast.addBackground(BG_RESOURCES);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0xfac973), 0, 10,
_missionComplete);
toast.addLabel("message", LabelFactory.createLabel(-1, 340, 23, 0xf0f0f0), 0, 50, _reward);
toast.addLabel("alloy", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR, 105,
22, 0xf0f0f0), 225, 104,
StringUtil.commaFormatNumber(missionInfo.alloyReward), TextFormatAlign.LEFT);
toast.addLabel("credit", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR, 105,
22, 0xf0f0f0), 75, 104,
StringUtil.commaFormatNumber(missionInfo.creditReward), TextFormatAlign.LEFT);
toast.addLabel("energy", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR,
105, 22, 0xf0f0f0), 75, 139,
StringUtil.commaFormatNumber(missionInfo.energyReward), TextFormatAlign.LEFT);
toast.addLabel("synthetic", LabelFactory.createLabel(LabelFactory.DYNAMIC_TEXT_COLOR,
105, 22, 0xf0f0f0), 225, 139,
StringUtil.commaFormatNumber(missionInfo.syntheticReward), TextFormatAlign.LEFT);
break;
}

case ToastEnum.FLEET_REPAIRED:
case ToastEnum.TRANSACTION_COMPLETE:
{
if (!event.prototype)
return;
toast.autoLayout = true;
toast.addBackground(BG_LARGE);
assetVO = getAssetVO(event.prototype);
addImageFromAsset("proto", assetVO, 0, 0);
toast.addTransactionLabels(event.transaction, assetVO, event.prototype);
break;
}
```

```
case ToastEnum.WRONG:
{
toast.addBackground(BG_SMALL);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0xaa3333), 0, 10, _alert);
label = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, 200, 24, 0xf0f0f0,
true);
label.setSize(250, 55);
label.leading = -6;
toast.addLabel("message", label, -1, 45, event.nextString);
break;
}

case ToastEnum.ALLIANCE:
{
toast.addBackground(BG_SMALL);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0x79da62), 0, 10, _alliance);
label = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, 200, 24, 0xf0f0f0,
true);
label.setSize(250, 55);
label.leading = -6;
toast.addLabel("message", label, -1, 45, event.nextString);
break;
}

case ToastEnum.BLUEPRINT:
{
var blueprint:BlueprintVO = BlueprintVO(event.prototype);
if (blueprint == null)
return;

toast.autoLayout = true;
toast.addBackground(BG_LARGE);
assetVO = getAssetVO(blueprint);
addImageFromAsset("proto", assetVO, 0, 0);
toast.addLabel("title", LabelFactory.createLabel(-1, 225, 32, 0xfac973), 130, 10, _blueprintTitle,
TextFormatAlign.LEFT);
toast.addLabel("message", LabelFactory.createLabel(-1, 250, 42, 0xf0f0f0, true), 130, 48,
event.nextString, TextFormatAlign.LEFT);
break;
}

case ToastEnum.BUBBLE_ALERT:
{
toast.addBackground(BG_LARGE);
assetVO = getAssetVO(event.prototype);
toast.addLabel("title", LabelFactory.createLabel(-1, 225, 32, 0xfac973), 130, 10,
event.nextString, TextFormatAlign.LEFT);
toast.addLabel("message", LabelFactory.createLabel(-1, 250, 42, 0xf0f0f0, true), 130, 48,
event.nextString,
```

```
TextFormatAlign.LEFT);
addImageFromAsset("proto", assetVO, 0, 0);
break;
}

case ToastEnum.BASE_RELOCATED:
{
toast.addBackground(BG_SMALL);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0x79da62), 0, 10, event.nextString);
label = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, 200, 24, 0xf0f0f0,
true);
label.setSize(250, 55);
label.leading = -6;
toast.addLabel("message", label, -1, 45, event.nextString);
break;
}

case ToastEnum.ACHIEVEMENT:
{
assetVO = getAssetVO(event.prototype);
toast.autoLayout = true;
toast.addBackground(BG_LARGE);
toast.addLabel("title", LabelFactory.createLabel(-1, 340, 32, 0x79da62), 25, 10,
_achievementUnlocked);
label = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, 200, 24, 0xf0f0f0,
true);
label.setSize(250, 55);
label.leading = -6;
toast.addLabel("message", label, 25 + (340 - 250) * 0.5, 45, assetVO.visibleName);
addImageFromAsset("proto", assetVO, -50, 0);
break;
}

default:
{
throw new Error("Tried to create a Toast with an invalid type");
break;
}
}

toastController.addToast(event.toastType, toast);
}

private function getAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName || assetName == '')
assetName = prototype.asset;
return assetModel.getEntityData(assetName);
}

private
```

```
function addImage( id:String, url:String, x:Number, y:Number, width:Number = 126,
height:Number = 126 ):ImageComponent
{
var image:ImageComponent = new ImageComponent();
image.init(width, height);
image.center = true;
image.x = x;
image.y = y;
toast.addImage(id, image);
assetModel.getFromCache(url, image.onImageLoaded, LoadPriority.LOW);
return image;
}

private function addImageFromAsset( id:String, assetVO:AssetVO, x:Number, y:Number,
width:Number = 126, height:Number = 126 ):ImageComponent
{
var url:String = "assets/" + ((assetVO.mediumImage && assetVO.mediumImage != "" &&
assetVO.mediumImage != "Default.png") ? assetVO.mediumImage : assetVO.smallImage)
var image:ImageComponent = addImage(id, url, x, y, width, height);
return image;
}
}
}
```

File 17: igw\com\controller\command\TransitionCommand.as

```
package com.controller.command
{
import com.event.TransitionEvent;
import com.ui.TransitionView;

import org.parade.core.ViewController;
import org.parade.core.ViewEvent;
import org.robotlegs.extensions.presenter.impl.Command;

public class TransitionCommand extends Command
{
[Inject]
public var event:TransitionEvent;
[Inject]
public var viewController:ViewController;

override public function execute():void
{
if (event.type == TransitionEvent.TRANSITION_BEGIN)
{
var view:TransitionView = TransitionView(viewController.getView(TransitionView));
if (!view)
{
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass
```

```
= TransitionView;
dispatch(viewEvent);
} else
view.resetEvents();
}
}
}
}
```

File 18: igw\com\controller\command\TransitionCoreCommand.as

```
package com.controller.command
{
import com.event.TransitionEvent;
import com.presenter.shared.ITransitionPresenter;

import org.robotlegs.extensions.presenter.impl.Command;

public class TransitionCoreCommand extends Command
{
[Inject]
public var transitionPresenter:ITransitionPresenter;

[Inject]
public var event:TransitionEvent;

override public function execute():void
{
if (event.type == TransitionEvent.TRANSITION_BEGIN)
transitionPresenter.addEvents(event.initEvent, event.cleanupEvent);
else if (event.type == TransitionEvent.TRANSITION_FAILED)
transitionPresenter.failed = true;
else
transitionPresenter.transitionComplete();
}

}
}
```

File 19: igw\com\controller\command\WelcomeBackCommand.as

```
package com.controller.command
{
import com.Application;
import com.controller.fte.FTEController;
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.TypeEnum;
import com.event.TransactionEvent;
import com.model.blueprint.BlueprintVO;
import
```

```
com.model.motd.MotDModel;
import com.model.player.CurrentUser;
import com.model.player.OfferVO;
import com.model.prototype.IPrototype;
import com.model.starbase.StarbaseModel;
import com.model.starbase.TradeRouteVO;
import com.presenter.starbase.IStarbasePresenter;
import com.presenter.starbase.ITradePresenter;
import com.ui.modal.building.RepairBaseView;
import com.ui.modal.construction.ConstructionInfoView;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.fullscreenprompt.FullScreenPromptModal;
import com.ui.modal.information.BaseActionPromptModal;
import com.ui.modal.information.MessageOfTheDayView;

import com.ui.modal.offers.OfferView;
import org.parade.core.IView;
import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;
import org.parade.util.DeviceMetrics;
import org.robotlegs.extensions.presenter.impl.Command;

import com.model.player.CurrentUser;

import com.model.prototype.PrototypeModel;

public class WelcomeBackCommand extends Command
{
public static var shownMOTD:Boolean = false;
public static var shownPromptToAction:Boolean = false;
public static var shouldShowPromptToFullScreen:Boolean = false;
public static var shownPromptToFullScreen:Boolean = false;
public static var shownDailyOffer:Boolean = false;

[Inject]
public var fteController:FTEController;
[Inject]
public var motdModel:MotDModel;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var starbasePresenter:IStarbasePresenter;
[Inject]
public var tradePresenter:ITradePresenter;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var viewFactory:IViewFactory;

override public function execute():void
{
```

```
shouldShowPromptToFullScreen = (DeviceMetrics.WIDTH_PIXELS <
DeviceMetrics.MAX_WIDTH_PIXELS && DeviceMetrics.HEIGHT_PIXELS <
DeviceMetrics.MAX_HEIGHT_PIXELS);

if (fteController.running)
return;

if (starbaseModel.entryView)
showEntryView();

if (shouldShowPromptToFullScreen && !shownPromptToFullScreen)
showPromptToFullScreen();
else if (!shownMOTD)
showMOTD();
else if (!shownPromptToAction)
showPromptToAction();

//showBaseRepair();

if (!shownDailyOffer)
showDailyOffer();
}

private function showEntryView():void
{
var data:* = starbaseModel.entryData;
var view:IView = viewFactory.createView(starbaseModel.entryView);
switch (starbaseModel.entryView)
{
case ConstructionInfoView:
ConstructionInfoView(view).setup(data.type, data.proto);
break;
case ConstructionView:
ConstructionView(view).openOn(data.type, data.groupID, data.subItemID);
break;
}
starbaseModel.entryData = null;
starbaseModel.entryView = null;
viewFactory.notify(view);
}

private function showMOTD():void
{
if (motdModel.hasMessage)
{
shownMOTD = true;
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = MessageOfTheDayView;
dispatch(viewEvent);
}
```

```
else if (!shownPromptToAction)
showPromptToAction();
}

private function showPromptToAction():void
{
if (!shownPromptToAction)
{
shownPromptToAction = true;

var debug:Boolean = false;
if (debug || (CurrentUser.level <= 20 && !CurrentUser.hasPromptedForBuildAction))
{
CurrentUser.hasPromptedForBuildAction = true;

var actionPrompts:uint;

if (hasPendingBuilds())
actionPrompts |= BaseActionPromptModal.BUILD_ACTION;

if (hasPendingTrade())
actionPrompts |= BaseActionPromptModal.TRADE_ACTION;

var researchType:String = getAvailResearchByType();

if (researchType == TypeEnum.WEAPONS_FACILITY)
actionPrompts |= BaseActionPromptModal.RESEARCH_WEAPONS_ACTION;

else if (researchType == TypeEnum.DEFENSE_DESIGN)
actionPrompts |= BaseActionPromptModal.RESEARCH_DEFENSE_ACTION;

else if (researchType == TypeEnum.ADVANCED_TECH)
actionPrompts |= BaseActionPromptModal.RESEARCH_TECH_ACTION;

else if (researchType == TypeEnum.SHIPYARD)
actionPrompts |= BaseActionPromptModal.RESEARCH_SHIPS_ACTION;

if (actionPrompts == 0)
{
showPromptToFullScreen();
return;
}

var view:BaseActionPromptModal =
BaseActionPromptModal(viewFactory.createView(BaseActionPromptModal));
view.actionTypes = actionPrompts;
viewFactory.notify(view);
} else
showPromptToFullScreen();
} else
showPromptToFullScreen();
```

```
}

private function showPromptToFullScreen():void
{
if (shouldShowPromptToFullScreen && !shownPromptToFullScreen)
{
shownPromptToFullScreen = true;
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = FullScreenPromptModal;
dispatch(viewEvent);
}
}

private function showBaseRepair():void
{
if (starbaseModel.isBaseDamaged() && !RepairBaseView.PLAYER_KNOWS_ABOUT_REPAIR
&& transactionController.getBaseRepairTransaction() == null)
{
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = RepairBaseView;
dispatch(viewEvent);
}
}

private function showDailyOffer():void
{
if (!shownDailyOffer)
{
shownDailyOffer = true;

var offers:Vector.<OfferVO> = CurrentUser.offers;
if (offers.length > 0)
{
var currentOffer:OfferVO;
currentOffer = offers[0];
var offerDurationMS:Number = currentOffer.offerDuration * 3600 * 1000;
if(!(currentOffer.offerPrototype == "BeginnerOffer" && (offerDurationMS -
currentOffer.timeRemainingMS) <
PrototypeModel.instance.getConstantPrototypeValueByName("WelcomeBackBeginnerOfferHideTimeMilli
{
var offerView:OfferView = OfferView(viewFactory.createView(OfferView));
offerView.offerProtoName = currentOffer;
viewFactory.notify(offerView);
}
}
}
}

private function hasPendingBuilds():Boolean
{
```

```
return transactionController.getAllStarbaseBuildingTransactions().length == 0;
}

private function hasPendingTrade():Boolean
{
var tradeVO:TradeRouteVO;
var routes:Vector.<TradeRouteVO> = tradePresenter.tradeRoutes;
var crntTradeRoutes:int;
var maxTradeRoutes:int = tradePresenter.maxUnlockedContracts;

for (var i:int; i < routes.length; i++)
{
tradeVO = routes[i];

if (tradeVO.end != 0)
crntTradeRoutes++;
}

if (crntTradeRoutes < maxTradeRoutes)
return true;

return false;
}

private function getAvailResearchByType():String
{
var type:String;
var src:Vector.<IPrototype> = starbasePresenter.researchPrototypes;
var typeHash:Object = {};

for (var proto:IPrototype, rqmt:RequirementVO, bprt:BlueprintVO, i:int; i < src.length; i++)
{
proto = src[i];

//filter
if ((proto.getValue('requiredFaction') == CurrentUser.faction || proto.getValue('requiredFaction')
== '') && !proto.getValue('hideWhileLocked'))
{
rqmt = starbasePresenter.getRequirements(TransactionEvent.STARBASE_RESEARCH, proto);
bprt = starbasePresenter.getBlueprintByName(proto.name);

if (rqmt.allMet || (bprt && !bprt.complete && bprt.partsCollected != 0))
{
if (rqmt.purchaseVO.canPurchase || rqmt.purchaseVO.canPurchaseResourcesWithPremium ||
rqmt.purchaseVO.canPurchaseWithPremium)
{
type = proto.getValue("requiredBuildingClass");

if (type == TypeEnum.WEAPONS_FACILITY)
typeHash[TypeEnum.WEAPONS_FACILITY]
```

```
= true;

else if (type == TypeEnum.DEFENSE_DESIGN)
typeHash[TypeEnum.DEFENSE_DESIGN] = true;

else if (type == TypeEnum.ADVANCED_TECH)
typeHash[TypeEnum.ADVANCED_TECH] = true;

else if (type == TypeEnum.SHIPYARD)
typeHash[TypeEnum.SHIPYARD] = true;
}
}
}
}

if (typeHash.hasOwnProperty(TypeEnum.WEAPONS_FACILITY))
return TypeEnum.WEAPONS_FACILITY;

else if (typeHash.hasOwnProperty(TypeEnum.DEFENSE_DESIGN))
return TypeEnum.DEFENSE_DESIGN;

else if (typeHash.hasOwnProperty(TypeEnum.ADVANCED_TECH))
return TypeEnum.ADVANCED_TECH;

else if (typeHash.hasOwnProperty(TypeEnum.SHIPYARD))
return TypeEnum.SHIPYARD;

else
return "";
}
}
}

-------------------------------------
File 20: igw\com\controller\command\account\ConnectionCommand.as
package com.controller.command.account
{
import com.Application;
import com.controller.ServerController;
import com.controller.fte.FTEController;
import com.enum.TimeLogEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.enum.ui.ButtonEnum;
import com.event.BattleEvent;
import com.event.PaywallEvent;
import com.event.SectorEvent;
import com.event.ServerEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import
```

```
com.event.TransitionEvent;
import com.model.player.CurrentUser;
import com.model.starbase.StarbaseModel;
import com.presenter.preload.PreloadPresenter;
import com.presenter.shared.ITransitionPresenter;
import com.service.ExternalInterfaceAPI;
import com.service.server.connections.Connection;
import com.service.server.connections.DevConnection;
import com.service.server.connections.FacebookConnection;
import com.service.server.connections.KabamConnection;
import com.service.server.connections.KongregateConnection;
import com.service.server.connections.XsollaConnection;
import com.service.server.connections.SteamConnection;
import com.service.server.connections.GuestConnection;
import com.service.server.outgoing.proxy.ClientLoginRequest;
import com.ui.alert.ConfirmationView;
import com.ui.core.ButtonPrototype;
import com.ui.core.ViewFactory;
import com.util.TimeLog;

import flash.events.Event;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;
import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;
import org.robotlegs.extensions.presenter.impl.Command;

public class ConnectionCommand extends Command
{
[Inject]
public var event:ServerEvent;
[Inject]
public var fteController:FTEController;
[Inject]
public var presenter:ITransitionPresenter;
[Inject]
public var serverController:ServerController;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var viewFactory:IViewFactory;

private static const _logger:ILogger = getLogger('ConnectionCommand');

override public function execute():void
{
//ExternalInterfaceAPI.logConsole("Imperium Connection Init");
Application.CONNECTION_STATE
```

```
= event.type;
//ExternalInterfaceAPI.logConsole("Imperium Connection State: " +
Application.CONNECTION_STATE);
_logger.info("execute - Application.CONNECTION_STATE = {0}", [event.type]);
switch (event.type)
{
case ServerEvent.CONNECT_TO_PROXY:
if (Application.NETWORK == Application.NETWORK_KABAM || Application.NETWORK ==
Application.NETWORK_DEV)
establishConnection(KabamConnection);
else if (Application.NETWORK == Application.NETWORK_KONGREGATE)
establishConnection(KongregateConnection);
else if (Application.NETWORK == Application.NETWORK_STEAM)
establishConnection(SteamConnection);
else if (Application.NETWORK == Application.NETWORK_GUEST)
establishConnection(GuestConnection);
else if (Application.NETWORK == Application.NETWORK_XSOLLA)
{
if (CONFIG::FLASH_DEBUG_MODE == true || CONFIG::FLASH_LIVE_DEBUG_MODE)
establishConnection(DevConnection);
else
establishConnection(XsollaConnection);
}
else if (Application.NETWORK == Application.NETWORK_FACEBOOK)
establishConnection(FacebookConnection);
else if (CurrentUser.naid)
establishConnection(DevConnection);
else
Application.CONNECTION_STATE = ServerEvent.NOT_CONNECTED;

break;

case ServerEvent.LOGIN_TO_ACCOUNT:
TimeLog.startTimeLog(TimeLogEnum.LOGIN_TO_ACCOUNT);
var proxyLogin:ClientLoginRequest =
ClientLoginRequest(serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_LOGIN));
proxyLogin.name = CurrentUser.id;
proxyLogin.challengeToken = CurrentUser.authID; // this should be given to us by KABAM
authentication
proxyLogin.clientVersion = 1; // TODO - increment this as we break binary compatibility

//ExternalInterfaceAPI.logConsole("Imperium Credentails: id = " + CurrentUser.id + " ; auth = " +
CurrentUser.authID);
serverController.send(proxyLogin);
serverController.lockRead = true;
break;

case ServerEvent.NEED_CHARACTER_CREATE:
dispatch(new StateEvent(StateEvent.CREATE_CHARACTER));
break;
```

```
case ServerEvent.BANNED:
case ServerEvent.SUSPENSION:
case ServerEvent.MAINTENANCE:
case ServerEvent.FAILED_TO_CONNECT:
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_FAILED);
dispatch(transitionEvent);


var title:String;
var body:String;

if (event.type == ServerEvent.BANNED)
{
title = "ACCOUNT BANNED";
body = "This Account has been banned. If you think this is wrong please contact support.";
} else if (event.type == ServerEvent.SUSPENSION)
{
title = "ACCOUNT SUSPENSION";
body = "This Account has been suspended. If you think this is wrong please contact support.";
} else if (event.type == ServerEvent.MAINTENANCE)
{
title = "MAINTENANCE";
body = "The game is currently undergoing maintenance. Please try again later.";
} else
{
title = "CONNECTION FAILED";
body = "Failed to connect. Try to refresh and if that does not work please contact support.";
ExternalInterfaceAPI.refresh();
break;
}
ExternalInterfaceAPI.logConsole("Failed to connect: " + title + " - " + body);

var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype('OK', null, null, true, ButtonEnum.RED_A));

var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
var view:ConfirmationView = ConfirmationView(viewFactory.createView(ConfirmationView));

view.setup(title, body, buttons)
viewEvent.targetView = view;
dispatch(viewEvent);

break;
default:
sendPixelRequest();
onAuthorized();
break;
}
```

```
if (presenter)
presenter.updateView();
}

private function onAuthorized():void
{
TimeLog.endTimeLog(TimeLogEnum.LOGIN_TO_ACCOUNT);
//If the preloader is done then we want to notify the loading screen that we're switching views
if (PreloadPresenter.complete)
{
var event:Event;
if (fteController.startInSector)
event = new SectorEvent(SectorEvent.CHANGE_SECTOR,
starbaseModel.homeBase.sectorID);
else if (starbaseModel.homeBase.battleServerAddress != null)
event = new BattleEvent(BattleEvent.BATTLE_JOIN,
starbaseModel.homeBase.battleServerAddress);
else if (starbaseModel.homeBase.instancedMissionAddress != null)
event = new BattleEvent(BattleEvent.BATTLE_JOIN,
starbaseModel.homeBase.instancedMissionAddress);
else
event = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(event);

if (Application.NETWORK == Application.NETWORK_KONGREGATE || Application.NETWORK
== Application.NETWORK_FACEBOOK)
{
var paywall:PaywallEvent = new PaywallEvent(PaywallEvent.BUY_ITEM);
dispatch(paywall);
}
}
}

private function sendPixelRequest():void
{
if (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER)
ExternalInterfaceAPI.popPixel(1);
}

private function establishConnection( ConnectionClass:Class ):void
{
//ExternalInterfaceAPI.logConsole("connecting...");
var connection:Connection = new ConnectionClass();
injector.injectInto(connection);
connection.connect();
}
}
}

-----------------------------------
```

File 21: igw\com\controller\command\account\MATCommand.as

```
package com.controller.command.account
{
import com.freshplanet.ane.AirDeviceId;
import com.hasoffers.nativeExtensions.MobileAppTracker;

import flash.desktop.NativeApplication;
import flash.filesystem.File;
import flash.filesystem.FileMode;
import flash.filesystem.FileStream;

import org.robotlegs.extensions.presenter.impl.Command;

/**
* Used in the mobile version of the game.
* Tracks app installs and updates with kabam.com
*/
public class MATCommand extends Command
{
private var _file:File;
override public function execute():void
{
if (!AirDeviceId.getInstance().isOnDevice)
return;

MobileAppTracker.instance.init("885", "429151a2d9a0e1601b518a5cda19c750");

//open the mat storage file
_file = File.applicationStorageDirectory.resolvePath("mat.txt");
if (!_file.exists)
trackInstall();
else
trackUpdate();
}

private function trackInstall():void
{
//get the app version number and save it out into a file
var fileStream:FileStream = new FileStream();
fileStream.open(_file, FileMode.WRITE);
fileStream.position = 0;
fileStream.writeUTF(appVersionNumber);
fileStream.close();

MobileAppTracker.instance.trackInstall();
}

private function trackUpdate():void
{
//compare
```

```
the version numbers. if they differ, track an update
var fileStream:FileStream = new FileStream();
fileStream.open(_file, FileMode.UPDATE);
fileStream.position = 0;
if (fileStream.readUTF() != appVersionNumber)
{
fileStream.position = 0;
fileStream.writeUTF(appVersionNumber);
MobileAppTracker.instance.trackUpdate();
}
fileStream.close();
}

private function get appVersionNumber():String
{
var appXML:XML = NativeApplication.nativeApplication.applicationDescriptor;
var ns:Namespace = appXML.namespace();
return "v" + appXML.ns::versionNumber;
}
}
}
```

File 22: igw\com\controller\command\account\PaymentCommand.as

```
package com.controller.command.account
{
import com.Application;
import com.controller.ServerController;
import com.controller.fte.FTEController;
import com.enum.TimeLogEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.enum.ui.ButtonEnum;
import com.event.BattleEvent;
import com.event.PaywallEvent;
import com.event.SectorEvent;
import com.event.ServerEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.TransitionEvent;
import com.model.player.CurrentUser;
import com.model.starbase.StarbaseModel;
import com.presenter.preload.PreloadPresenter;
import com.presenter.shared.ITransitionPresenter;
import com.service.ExternalInterfaceAPI;
import com.service.server.connections.Connection;
import com.service.server.connections.XsollaPaymentConnection;
import com.service.server.outgoing.proxy.ClientLoginRequest;
import com.ui.alert.ConfirmationView;
import com.ui.core.ButtonPrototype;
import com.ui.core.ViewFactory;
import
```

```
com.util.TimeLog;

import flash.events.Event;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;
import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;
import org.robotlegs.extensions.presenter.impl.Command;

public class PaymentCommand extends Command
{
//[Inject]
//public var event:ServerEvent;
//[Inject]
//public var fteController:FTEController;
//[Inject]
public var presenter:ITransitionPresenter;
//[Inject]
//public var serverController:ServerController;
//[Inject]
//public var starbaseModel:StarbaseModel;
//[Inject]
//public var viewFactory:IViewFactory;
private static const _logger:ILogger = getLogger('PaymentCommand');

override public function execute():void
{
establishConnection(XsollaPaymentConnection);

if (presenter)
presenter.updateView();
}

private function establishConnection( ConnectionClass:Class ):void
{
//ExternalInterfaceAPI.logConsole("connecting...");
var connection:Connection = new ConnectionClass();
injector.injectInto(connection);
connection.connect();
}
}
}

------------------------------------
File 23: igw\com\controller\command\load\LoadCompleteCommand.as
package com.controller.command.load
{
import
```

```
com.controller.sound.SoundController;
import com.enum.TimeLogEnum;
import com.event.LoadEvent;
import com.model.asset.AssetModel;
import com.model.asset.ISpriteSheet;
import com.model.prototype.PrototypeModel;
import com.service.loading.LoadingTypes;
import com.service.loading.loaditems.BatchLoadItem;
import com.service.loading.loaditems.ILoadItem;
import com.util.TimeLog;

import flash.display.Bitmap;
import flash.media.Sound;

import org.robotlegs.extensions.presenter.impl.Command;

public class LoadCompleteCommand extends Command
{
[Inject]
public var assetModel:AssetModel;
[Inject]
public var event:LoadEvent;
[Inject]
public var prototypeModel:PrototypeModel;
[Inject]
public var soundController:SoundController;

override public function execute():void
{
var loadItem:ILoadItem = event.loadItem;
var asset:Object = loadItem.asset;
var batchLoad:BatchLoadItem;
var i:int;
var items:Vector.<ILoadItem>;
switch (loadItem.type)
{
case LoadingTypes.BITMAP:
asset = Bitmap(loadItem.asset).bitmapData;
assetModel.cache(loadItem.url, asset);
break;
case LoadingTypes.MESH:
break;
case LoadingTypes.SOUND:
asset = loadItem.asset;
soundController.addAudio(loadItem.url, Sound(asset));
break;
case LoadingTypes.SWF:
assetModel.cache(loadItem.url, asset);
break;
case LoadingTypes.TEXT:
try
```

```
{
TimeLog.startTimeLog(TimeLogEnum.JSON_PARSE, loadItem.url);
asset = JSON.parse(String(loadItem.asset));
TimeLog.endTimeLog(TimeLogEnum.JSON_PARSE, loadItem.url);
}
catch( e:Error )
{
asset = AssetModel.FAILED;
assetModel.cache(loadItem.url, asset);
TimeLog.endTimeLog(TimeLogEnum.JSON_PARSE, loadItem.url);
break;
}
//we know what this data is, so skip caching it
if (asset.format == "Not Loc")
{
//assetModel.addGameAssetData(asset);
} else
{
if (asset.ShipPrototypes || asset.EventPrototypes)
prototypeModel.addPrototypeData(asset);
else
assetModel.cache(loadItem.url, asset); //dont know what type it is so cache for now
}
break;
case LoadingTypes.XML:
assetModel.cache(loadItem.url, XML(asset));
break;
case LoadingTypes.BATTLEREPLAY:
assetModel.cache(loadItem.url, asset);
break;
case LoadingTypes.SPRITE_SHEET:
batchLoad = BatchLoadItem(loadItem);
items = batchLoad.items;
for (i = 0; i < items.length; i += 2)
{
assetModel.removeFromCache(items[i].url);
assetModel.removeFromCache(items[i + 1].url);
assetModel.initSpriteSheet(items[i].url, Bitmap(items[i].asset).bitmapData, XML(items[i +
1].asset));
}
break;
case LoadingTypes.SPRITE_SHEET_MESH:
batchLoad = BatchLoadItem(loadItem);
items = batchLoad.items;
for (i = 0; i < items.length; i++)
{
assetModel.removeFromCache(items[i].url);
assetModel.initSpriteSheet(items[i].url, items[i].asset, null);
}
break;
```

```
}
}
}
}

------------------------------------
File 24: igw\com\controller\command\load\RequestLoadCommand.as
package com.controller.command.load
{
import com.event.RequestLoadEvent;
import com.service.loading.ILoadService;

import org.robotlegs.extensions.presenter.impl.Command;

public class RequestLoadCommand extends Command
{
[Inject]
public var event:RequestLoadEvent;

[Inject]
public var loadService:ILoadService;

override public function execute():void
{
if (event.url != null)
{
loadService.lazyLoad(event.url, event.priority, true, event.absoluteURL);
}
if (event.urls != null)
{
loadService.loadBatch(event.batchType, event.urls, event.priority);
}
}
}
}

------------------------------------
File 25: igw\com\controller\command\state\BattleCoreCommand.as
package com.controller.command.state
{
import com.Application;
import com.event.BattleEvent;
import com.event.StateEvent;
import com.event.TransitionEvent;
import com.model.battle.BattleModel;
import com.model.scene.SceneModel;
import com.model.sector.SectorModel;

import org.parade.core.ViewEvent;
import
```

```
org.robotlegs.extensions.presenter.impl.Command;

public class BattleCoreCommand extends Command
{
[Inject]
public var battleModel:BattleModel;
[Inject]
public var event:BattleEvent;
[Inject]
public var sceneModel:SceneModel;
[Inject]
public var sectorModel:SectorModel;

override public function execute():void
{
var viewEvent:ViewEvent;
switch (event.type)
{
case BattleEvent.BATTLE_JOIN:
case BattleEvent.BATTLE_REPLAY:
battleModel.isReplay = (event.type == BattleEvent.BATTLE_REPLAY);
battleModel.finished = false;
battleModel.battleServerAddress = event.battleServerAddress;
if (Application.STATE != StateEvent.GAME_BATTLE)
{
battleModel.oldGameState = Application.STATE;
//save the view location so we can center on the correct spot again once the battle is over
if (Application.STATE == StateEvent.GAME_SECTOR)
{
battleModel.oldSector = sectorModel.sectorID;
battleModel.focusLocation.setTo(sceneModel.focus.x, sceneModel.focus.y);
} else
battleModel.focusLocation.setTo(0, 0);
}
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
var cleanupState:String;
switch (Application.STATE)
{
case StateEvent.GAME_BATTLE_INIT:
case StateEvent.GAME_BATTLE:
cleanupState = StateEvent.GAME_BATTLE_CLEANUP;
break;
case StateEvent.GAME_SECTOR_INIT:
case StateEvent.GAME_SECTOR:
cleanupState = StateEvent.GAME_SECTOR_CLEANUP;
break;
case StateEvent.GAME_STARBASE:
cleanupState = StateEvent.GAME_STARBASE_CLEANUP;
break;
case
```

```
null:
case StateEvent.PRELOAD:
cleanupState = "";
break;
default:
cleanupState = StateEvent.DEFAULT_CLEANUP;
break;
}
transitionEvent.addEvents(new StateEvent(StateEvent.GAME_BATTLE_INIT, cleanupState),
new StateEvent(cleanupState, StateEvent.GAME_BATTLE_INIT));
dispatch(transitionEvent);
break;
}
}
}
}
```

File 26: igw\com\controller\command\state\CreateCharacterCommand.as

```
package com.controller.command.state
{
import com.ui.TransitionView;
import com.ui.modal.intro.FactionSelectView;

import org.parade.core.ViewController;
import org.parade.core.ViewEvent;

public class CreateCharacterCommand extends StateCommand
{
[Inject]
public var viewController:ViewController;

override public function execute():void
{
var view:TransitionView = TransitionView(viewController.getView(TransitionView));
if (view)
view.destroy();

var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = FactionSelectView;
dispatch(viewEvent);
}
}
}
```

File 27: igw\com\controller\command\state\GameCommand.as

```
package com.controller.command.state
{
```

```
import com.controller.fte.FTEController;
import com.controller.transaction.TransactionController;
import com.enum.PriorityEnum;
import com.event.MissionEvent;
import com.event.StateEvent;
import com.game.entity.systems.battle.TrailFXSystem;
import com.game.entity.systems.shared.VCSystem;
import com.game.entity.systems.shared.render.RenderSystem;
import com.game.entity.systems.starbase.StateSystem;
import com.model.battle.BattleModel;
import com.model.mission.MissionModel;
import com.model.motd.MotDModel;
import com.model.player.CurrentUser;
import com.model.starbase.StarbaseModel;
import com.ui.hud.battle.BattleShipSelectionView;
import com.ui.hud.battle.BattleUserView;
import com.ui.hud.battle.BattleView;
import com.ui.hud.sector.SectorView;
import com.ui.hud.shared.ChatView;
import com.ui.hud.shared.IconDrawerView;
import com.ui.hud.shared.MiniMapView;
import com.ui.hud.shared.PlayerView;
import com.ui.hud.shared.bridge.BridgeView;
import com.ui.hud.shared.command.CommandView;
import com.ui.hud.shared.engineering.EngineeringView;
import com.ui.hud.starbase.StarbaseView;

import org.ash.core.Game;
import org.parade.core.IView;
import org.parade.core.IViewFactory;
import org.parade.core.IViewStack;
import org.parade.core.ViewController;
import org.parade.core.ViewEvent;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class GameCommand extends StateCommand
{
[Inject]
public var battleModel:BattleModel;
[Inject]
public var fteController:FTEController;
[Inject]
public var game:Game;
[Inject]
public var missionModel:MissionModel;
[Inject]
public var motdModel:MotDModel;
[Inject]
public
```

```
var starbaseModel:StarbaseModel;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var viewController:ViewController;
[Inject]
public var viewFactory:IViewFactory;
[Inject]
public var viewStack:IViewStack;

override public function execute():void
{
var missionEvent:MissionEvent;
switch (event.type)
{
case StateEvent.GAME_BATTLE:
initBattle();
dispatchViewEvent(BattleUserView);
dispatchViewEvent(MiniMapView, true);
dispatchViewEvent(BattleView);
if (!battleModel.isInstancedMission && battleModel.participants.indexOf(CurrentUser.id) > -1 &&
(!battleModel.isBaseCombat || battleModel.baseOwnerID != CurrentUser.id))
dispatchViewEvent(BattleShipSelectionView);
dispatchViewEvent(ChatView, true);
//is this battle part of a mission? if so show the dialogue
if (!missionModel.currentMission.isFTE && battleModel.missionID ==
missionModel.currentMission.id)
{
missionEvent = new MissionEvent(MissionEvent.MISSION_SITUATIONAL);
dispatch(missionEvent);
}
break;
case StateEvent.GAME_STARBASE:
dispatchViewEvent(PlayerView, true);
dispatchViewEvent(StarbaseView);
dispatchViewEvent(MiniMapView, true);
dispatchViewEvent(BridgeView, true);
dispatchViewEvent(EngineeringView, true);
dispatchViewEvent(ChatView, true);
dispatchViewEvent(CommandView, true);
dispatchViewEvent(IconDrawerView, true);
initStarbase();
break;
case StateEvent.GAME_SECTOR:
dispatchViewEvent(PlayerView, true);
dispatchViewEvent(MiniMapView, true);
dispatchViewEvent(EngineeringView, true);
dispatchViewEvent(SectorView);
dispatchViewEvent(BridgeView, true);
dispatchViewEvent(ChatView, true);
dispatchViewEvent(CommandView,
```

```
true);
dispatchViewEvent(IconDrawerView, true);
initSector();
break;
case StateEvent.GAME_BATTLE_CLEANUP:
if (!missionModel.currentMission.isFTE && battleModel.missionID ==
missionModel.currentMission.id && !battleModel.wonLastBattle)
{
missionEvent = new MissionEvent(MissionEvent.MISSION_FAILED);
dispatch(missionEvent);
} else if (!missionModel.currentMission.isFTE && missionModel.currentMission.complete &&
!missionModel.currentMission.rewardAccepted)
{
missionModel.missionComplete();
transactionController.dataImported();
}
case StateEvent.GAME_SECTOR_CLEANUP:
case StateEvent.GAME_STARBASE_CLEANUP:
viewStack.clearLayer(ViewEnum.BACKGROUND_LAYER);
viewStack.clearLayer(ViewEnum.GAME_LAYER);
break;
}
}

private function initBattle():void
{
game.addSystem(ObjectPool.get(RenderSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(VCSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(TrailFXSystem), PriorityEnum.RENDER);
}

private function initStarbase():void
{
game.addSystem(ObjectPool.get(RenderSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(VCSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(StateSystem), PriorityEnum.RESOLVE_COLLISIONS);
}

private function initSector():void
{
game.addSystem(ObjectPool.get(RenderSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(VCSystem), PriorityEnum.RENDER);
}

private function dispatchViewEvent( viewClass:Class, checkIfViewExists:Boolean = false ):void
{
if (checkIfViewExists)
{
var view:IView = viewController.getView(viewClass);
if (view != null)
{
```

```
//readd the view to maintain correct depth sorting
viewStack.addView(view);
return;
}
}
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = viewClass;
dispatch(viewEvent);
}
}
}



---
File 28: igw\com\controller\command\state\GameCoreCommand.as
package com.controller.command.state
{
import com.Application;
import com.controller.GameController;
import com.controller.ServerController;
import com.controller.fte.FTEController;
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.enum.FactionEnum;
import com.enum.PriorityEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.StateEvent;
import com.game.entity.systems.battle.ActiveDefenseSystem;
import com.game.entity.systems.battle.AreaSystem;
import com.game.entity.systems.battle.BeamSystem;
import com.game.entity.systems.battle.DebugLineSystem;
import com.game.entity.systems.battle.DroneSystem;
import com.game.entity.systems.battle.ShipSystem;
import com.game.entity.systems.battle.VitalsSystem;
import com.game.entity.systems.interact.BattleInteractSystem;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.sector.FleetSystem;
import com.game.entity.systems.shared.AnimationSystem;
import com.game.entity.systems.shared.FSMSystem;
import com.game.entity.systems.shared.MoveSystem;
import com.game.entity.systems.shared.TweenSystem;
import com.game.entity.systems.shared.background.BackgroundSystem;
import com.game.entity.systems.shared.grid.GridSystem;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.asset.AssetModel;
import com.model.asset.ISpritePack;
import
```

```
com.model.battle.BattleModel;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.scene.SceneModel;
import com.model.sector.SectorModel;
import com.model.starbase.StarbaseModel;
import com.presenter.battle.IBattlePresenter;
import com.presenter.sector.ISectorPresenter;
import com.presenter.shared.IGamePresenter;
import com.presenter.starbase.IStarbasePresenter;
import com.service.loading.ILoadService;
import com.service.server.incoming.data.BattleData;
import com.service.server.outgoing.proxy.ProxyConnectToBattleRequest;
import com.service.server.outgoing.proxy.ProxyConnectToSectorRequest;

import flash.system.System;
import flash.utils.Dictionary;

import org.ash.core.Game;
import org.ash.core.System;
import org.shared.ObjectPool;

public class GameCoreCommand extends StateCommand
{
private static var oldFaction:String;
private static var oldState:String;

[Inject]
public var assetModel:AssetModel;
[Inject]
public var battleModel:BattleModel;
[Inject]
public var fteController:FTEController;
[Inject]
public var game:Game;
[Inject]
public var gameController:GameController;
[Inject]
public var loadService:ILoadService;
[Inject]
public var playerModel:PlayerModel;
[Inject]
public var sceneModel:SceneModel;
[Inject]
public var sectorModel:SectorModel;
[Inject]
public var serverController:ServerController;
[Inject]
public var soundController:SoundController;
[Inject]
public
```

```
var starbaseModel:StarbaseModel;

public var presenter:IGamePresenter;

override public function execute():void
{
Application.STATE = event.type;
switch (event.type)
{
case StateEvent.GAME_BATTLE_INIT:
initBattle();
game.addSystem(ObjectPool.get(BattleInteractSystem), PriorityEnum.UPDATE);
serverController.protocolListener = ProtocolEnum.BATTLE_CLIENT;
if( !battleModel.isReplay )
{
//connect to the battle
var request:ProxyConnectToBattleRequest =
ProxyConnectToBattleRequest(serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_CONNECT_TO_BATTLE));
request.key = battleModel.battleServerAddress;
serverController.send(request);
}
else
{
serverController.requestReplay( battleModel.battleServerAddress );
}
break;
case StateEvent.GAME_BATTLE:
clearSpritePacks();
getPresenter();
presenter.loadBackground(battleModel, true);
gameController.presenter = presenter;
//play the battle music
if (!fteController.running)
soundController.playSound(AudioEnum.AFX_BG_BATTLE_MUSIC, 0.2, 0, 100);
BattleInteractSystem(game.getSystem(BattleInteractSystem)).init();
game.addSystem(ObjectPool.get(BattleInteractSystem), PriorityEnum.UPDATE);
break;
case StateEvent.GAME_STARBASE:
//set the current sector to that of the base so the player goes there when exiting the starbase
sectorModel.updateSector(starbaseModel.currentBase.sector);
sectorModel.focusFleetID = null;
clearSpritePacks();
initStarbase();
game.addSystem(ObjectPool.get(StarbaseInteractSystem), PriorityEnum.UPDATE);
getPresenter();
presenter.loadBackground(battleModel);
serverController.protocolListener = ProtocolEnum.STARBASE_CLIENT;
gameController.presenter = presenter;
if (!fteController.running)
{
```

```
switch (CurrentUser.faction)
{
case FactionEnum.IGA:
soundController.playSound(AudioEnum.AFX_BG_IGA_THEME, 0.17, 0, 100);
break;
case FactionEnum.SOVEREIGNTY:
soundController.playSound(AudioEnum.AFX_BG_SOVEREIGNTY_THEME, 0.18, 0, 100);
break;
case FactionEnum.TYRANNAR:
soundController.playSound(AudioEnum.AFX_BG_TYRANNAR_THEME, 0.14, 0, 100);
break;
}
}
break;
case StateEvent.GAME_SECTOR_INIT:
initSector();
game.addSystem(ObjectPool.get(SectorInteractSystem), PriorityEnum.UPDATE);
serverController.protocolListener = ProtocolEnum.SECTOR_CLIENT;
gameController.sectorRequestBaseline();
break;
case StateEvent.GAME_SECTOR:
clearSpritePacks();
getPresenter();
gameController.presenter = presenter;
// soundController.playSound(AudioEnum.AFX_BG_MAIN_THEME, 0.07, 0, 100);
presenter.loadBackground(battleModel);
if (!fteController.running)
{
switch (sectorModel.sectorFaction)
{
case FactionEnum.IGA:
//If the player is in their own faction space use the main them; otherwise, use the faction theme
if (CurrentUser.faction == sectorModel.sectorFaction)
soundController.playSound(AudioEnum.AFX_BG_MAIN_THEME, 0.07, 0, 100);
else
soundController.playSound(AudioEnum.AFX_BG_IGA_THEME, 0.17, 0, 100);
//Ambient Space Sounds
soundController.playSound(AudioEnum.AFX_STG_AMBIENT_IGA_SPACE_SOUNDS, 1, 0,
100);
break;
case FactionEnum.SOVEREIGNTY:
//If the player is in their own faction space use the main them; otherwise, use the faction theme
if (CurrentUser.faction == sectorModel.sectorFaction)
soundController.playSound(AudioEnum.AFX_BG_MAIN_THEME, 0.07, 0, 100);
else
soundController.playSound(AudioEnum.AFX_BG_SOVEREIGNTY_THEME, 0.18, 0, 100);
//Ambient Space Sounds

soundController.playSound(AudioEnum.AFX_STG_AMBIENT_SOVEREIGNTY_SPACE_SOUNDS,
1,
```

```
0, 100);
break;
case FactionEnum.TYRANNAR:
//If the player is in their own faction space use the main them; otherwise, use the faction theme
if (CurrentUser.faction == sectorModel.sectorFaction)
soundController.playSound(AudioEnum.AFX_BG_MAIN_THEME, 0.07, 0, 100);
else
soundController.playSound(AudioEnum.AFX_BG_TYRANNAR_THEME, 0.14, 0, 100);
//Ambient Space Sounds
soundController.playSound(AudioEnum.AFX_STG_AMBIENT_TYRANNAR_SPACE_SOUNDS,
1, 0, 100);
break;
}
}
break;
case StateEvent.GAME_BATTLE_CLEANUP:
{
battleModel.cleanup();
BattleData.globalInstance.destroy();
getPresenter();
cleanup();
serverController.cleanupBattle();
}
break;
case StateEvent.GAME_SECTOR_CLEANUP:
if (event.nextState != StateEvent.GAME_SECTOR)
{
var sectorDisconnect:ProxyConnectToSectorRequest =
ProxyConnectToSectorRequest(serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_CONNECT_TO_SECTOR));
serverController.send(sectorDisconnect);
}

switch (sectorModel.sectorFaction)
{
case FactionEnum.IGA:
soundController.stopSound(AudioEnum.AFX_STG_AMBIENT_IGA_SPACE_SOUNDS);
break;
case FactionEnum.SOVEREIGNTY:

soundController.stopSound(AudioEnum.AFX_STG_AMBIENT_SOVEREIGNTY_SPACE_SOUNDS);
break;
case FactionEnum.TYRANNAR:

soundController.stopSound(AudioEnum.AFX_STG_AMBIENT_TYRANNAR_SPACE_SOUNDS);
break;
}
getPresenter();
cleanup();
break;
case
```

```
StateEvent.DEFAULT_CLEANUP:
case StateEvent.GAME_STARBASE_CLEANUP:
getPresenter();
cleanup();
break;
}
presenter = null;
}

private function initBattle():void
{
game.addSystem(ObjectPool.get(AnimationSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(ActiveDefenseSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(AreaSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(BackgroundSystem), PriorityEnum.MOVE);
game.addSystem(ObjectPool.get(BeamSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(DroneSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(FSMSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(GridSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(MoveSystem), PriorityEnum.MOVE);
game.addSystem(ObjectPool.get(ShipSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(StarbaseSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(TweenSystem), PriorityEnum.RESOLVE_COLLISIONS);
if (CONFIG::DEBUG == true)
game.addSystem(ObjectPool.get(DebugLineSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(VitalsSystem), PriorityEnum.RENDER);
}

private function initStarbase():void
{
game.addSystem(ObjectPool.get(AnimationSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(BackgroundSystem), PriorityEnum.MOVE);
game.addSystem(ObjectPool.get(MoveSystem), PriorityEnum.MOVE);
game.addSystem(ObjectPool.get(FSMSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(GridSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(StarbaseSystem), PriorityEnum.UPDATE);
game.addSystem(ObjectPool.get(TweenSystem), PriorityEnum.RESOLVE_COLLISIONS);
}

private function initSector():void
{
game.addSystem(ObjectPool.get(AnimationSystem), PriorityEnum.RENDER);
game.addSystem(ObjectPool.get(BackgroundSystem), PriorityEnum.MOVE);
game.addSystem(ObjectPool.get(MoveSystem), PriorityEnum.MOVE);
game.addSystem(ObjectPool.get(FleetSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(GridSystem), PriorityEnum.RESOLVE_COLLISIONS);
game.addSystem(ObjectPool.get(TweenSystem), PriorityEnum.RESOLVE_COLLISIONS);
}

private function cleanup():void
{
```

```
ObjectPool.gc();

//remove the entities
game.removeAllEntities();
//pool the systems
var systems:Vector.<org.ash.core.System> = game.removeAllSystems();
for (var i:int = 0; i < systems.length; i++)
ObjectPool.give(systems[i]);
//stop any spritepacks that may be building
assetModel.stopAllSpritePackBuilds();

playerModel.removeAllPlayers();
if (presenter)
presenter.cleanup();
}

private function clearSpritePacks():void
{
if (oldState != null)
{
var factionsDiffer:Boolean = oldFaction != sectorModel.sectorFaction;
var spritePacks:Dictionary = assetModel.spritePacks;
var mask:int = stateMask;
var memoryUsage:Number = flash.system.System.totalMemory * 0.000000954;
//clear out old spritepacks
for each (var spritePack:ISpritePack in spritePacks)
{
if ((spritePack.usedBy != 8 && (spritePack.usedBy & mask) == 0) || (spritePack.usedBy == 8 &&
factionsDiffer))
{
assetModel.removeSpritePack(spritePack);
}
}
}
oldFaction = sectorModel.sectorFaction;
oldState = Application.STATE;
}

private function getPresenter():void
{
switch (event.type)
{
case StateEvent.GAME_STARBASE_CLEANUP:
presenter = gameController.presenter;
if (presenter && !(presenter is IStarbasePresenter))
presenter = null;
break;
case StateEvent.GAME_STARBASE:
presenter = injector.getInstance(IStarbasePresenter);
break;
```

```
case StateEvent.GAME_BATTLE_CLEANUP:
presenter = gameController.presenter;
if (presenter && !(presenter is IBattlePresenter))
presenter = null;
break;
case StateEvent.GAME_BATTLE:
presenter = injector.getInstance(IBattlePresenter);
break;
case StateEvent.GAME_SECTOR_CLEANUP:
presenter = gameController.presenter;
if (presenter && !(presenter is ISectorPresenter))
presenter = null;
break;
case StateEvent.GAME_SECTOR:
presenter = injector.getInstance(ISectorPresenter);
break;
}
}

private function get stateMask():int
{
if (event.type == StateEvent.GAME_BATTLE)
return 1;
else if (event.type == StateEvent.GAME_SECTOR)
return 2;
return 4;
}
}
}



---
File 29: igw\com\controller\command\state\PreloadCommand.as
package com.controller.command.state
{
import com.Application;
import com.controller.ServerController;
import com.controller.fte.FTEController;
import com.event.BattleEvent;
import com.event.SectorEvent;
import com.event.ServerEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.model.asset.ISpriteSheet;
import com.model.asset.SpriteSheet;
import com.model.asset.SpriteSheetStarling;
import com.model.starbase.StarbaseModel;
import com.presenter.preload.IPreloadPresenter;
import
```

```
com.ui.GameView;
import com.ui.PreloadView;

import flash.display.BitmapData;
import flash.events.Event;
import flash.utils.getDefinitionByName;

import org.ash.integration.swiftsuspenders.NodeLookup;
import org.parade.core.ViewEvent;
import org.starling.text.BitmapFont;
import org.starling.text.TextField;
import org.starling.textures.Texture;

public class PreloadCommand extends StateCommand
{
[Inject]
public var fteController:FTEController;
[Inject]
public var preloadPresenter:IPreloadPresenter;
[Inject]
public var serverController:ServerController;
[Inject]
public var starbaseModel:StarbaseModel;

override public function execute():void
{
if (event.type == StateEvent.PRELOAD)
begin();
else
complete();
}

private function begin():void
{
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = GameView;
dispatch(viewEvent);

viewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = PreloadView;
dispatch(viewEvent);
}

private function complete():void
{
//Used by Ash to determine which node type to use
NodeLookup.addLookup("IFleetNode", (!Application.STARLING_ENABLED) ?
"com.game.entity.nodes.sector.fleet.FleetNode" :
"com.game.entity.nodes.sector.fleet.FleetStarlingNode");
NodeLookup.addLookup("IShipNode", (!Application.STARLING_ENABLED) ?
"com.game.entity.nodes.battle.ship.ShipNode"
```

```
: "com.game.entity.nodes.battle.ship.ShipStarlingNode");
NodeLookup.addLookup("IVCNode", (!Application.STARLING_ENABLED) ?
"com.game.entity.nodes.shared.visualComponent.VCNode" :
"com.game.entity.nodes.shared.visualComponent.VCStarlingNode");
NodeLookup.addLookup("IVCSpriteNode", (!Application.STARLING_ENABLED) ?
"com.game.entity.nodes.shared.visualComponent.VCSpriteNode" :
"com.game.entity.nodes.shared.visualComponent.VCSpriteStarlingNode");

//set starling specific options
if (Application.STARLING_ENABLED)
{
injector.map(ISpriteSheet).toType(SpriteSheetStarling);

//set up the OpenSansBold bitmap font
var ac:Class = Class(getDefinitionByName('OpenSansBoldSpriteSheet'));
var bmd:BitmapData = BitmapData(new ac());
ac = Class(getDefinitionByName('FontsMain'))['OpenSansBoldXML'];
var xml:XML = XML(new ac());
var bf:BitmapFont = new BitmapFont(Texture.fromBitmapData(bmd, false), xml);
TextField.registerBitmapFont(bf, 'OpenSansBoldBitmap');

} else
{
injector.map(ISpriteSheet).toType(SpriteSheet);
}

preloadPresenter.completeSignal.dispatch();
serverController.lockRead = false;

if (Application.CONNECTION_STATE == ServerEvent.AUTHORIZED)
{
var event:Event;
if (fteController.startInSector)
event = new SectorEvent(SectorEvent.CHANGE_SECTOR,
starbaseModel.homeBase.sectorID);
else if (starbaseModel.homeBase.battleServerAddress != null)
event = new BattleEvent(BattleEvent.BATTLE_JOIN,
starbaseModel.homeBase.battleServerAddress);
else if (starbaseModel.homeBase.instancedMissionAddress != null)
event = new BattleEvent(BattleEvent.BATTLE_JOIN,
starbaseModel.homeBase.instancedMissionAddress);
else
event = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(event);
return;
}
}
}
}

-----------------------------------
```

File 30: igw\com\controller\command\state\SectorCoreCommand.as

```
package com.controller.command.state
{
import com.Application;
import com.controller.ChatController;
import com.event.SectorEvent;
import com.event.StateEvent;
import com.event.TransitionEvent;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.sector.SectorModel;
import com.model.starbase.StarbaseModel;

import org.robotlegs.extensions.presenter.impl.Command;

public class SectorCoreCommand extends Command
{
[Inject]
public var event:SectorEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var sectorModel:SectorModel;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var chatController:ChatController;

override public function execute():void
{
if (event.sector)
{
sectorModel.targetSector = event.sector;
}
if (event.focusFleetID)
{
var fleet:FleetVO = fleetModel.getFleet(event.focusFleetID);
if (fleet)
{
sectorModel.targetSector = fleet.sector != "" ? fleet.sector :
starbaseModel.getBaseByID(fleet.starbaseID).sectorID;
}
sectorModel.focusFleetID = event.focusFleetID;
}
if (sectorModel.targetSector == null || sectorModel.viewBase)
sectorModel.targetSector = starbaseModel.currentBase.sectorID;
sectorModel.focusLocation.setTo(event.focusX, event.focusY);
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
var
```

```
cleanupState:String;
switch (Application.STATE)
{
case StateEvent.GAME_BATTLE_INIT:
case StateEvent.GAME_BATTLE:
cleanupState = StateEvent.GAME_BATTLE_CLEANUP;
break;
case StateEvent.GAME_SECTOR_INIT:
case StateEvent.GAME_SECTOR:
cleanupState = StateEvent.GAME_SECTOR_CLEANUP;
break;
case StateEvent.GAME_STARBASE:
cleanupState = StateEvent.GAME_STARBASE_CLEANUP;
break;
case null:
case StateEvent.PRELOAD:
cleanupState = "";
break;
default:
cleanupState = StateEvent.DEFAULT_CLEANUP;
break;
}
transitionEvent.addEvents(new StateEvent(StateEvent.GAME_SECTOR_INIT, cleanupState),
new StateEvent(cleanupState, StateEvent.GAME_SECTOR_INIT));
dispatch(transitionEvent);
}
}
}
```

File 31: igw\com\controller\command\state\ShutdownCommand.as

```
package com.controller.command.state
{
import com.Application;
import com.event.StateEvent;
import com.ui.ReconnectView;

import org.parade.core.ViewEvent;

public class ShutdownCommand extends StateCommand
{
override public function execute():void
{
if (event.type == StateEvent.SHUTDOWN_FINISH)
finishShutdown();
}

private function finishShutdown():void
{
```

```
Application.PROXY_SERVER = null;
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetClass = ReconnectView;
dispatch(viewEvent);
}
}
}
```

File 32: igw\com\controller\command\state\ShutdownCoreCommand.as

```
package com.controller.command.state
{
import com.Application;
import com.controller.GameController;
import com.controller.ServerController;
import com.event.StateEvent;
import com.event.TransitionEvent;

import org.parade.core.ViewEvent;

public class ShutdownCoreCommand extends StateCommand
{
[Inject]
public var gameController:GameController;
[Inject]
public var serverController:ServerController;

override public function execute():void
{
if (event.type == StateEvent.SHUTDOWN_START)
startShutdown();
else
finishShutdown();
}

private function startShutdown():void
{
//close the server connection
serverController.disconnect();
gameController.disconnect();

var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
var cleanupState:String;
switch (Application.STATE)
{
case StateEvent.GAME_BATTLE_INIT:
case StateEvent.GAME_BATTLE:
cleanupState = StateEvent.GAME_BATTLE_CLEANUP;
break;
```

```
case StateEvent.GAME_SECTOR_INIT:
case StateEvent.GAME_SECTOR:
cleanupState = StateEvent.GAME_SECTOR_CLEANUP;
break;
case StateEvent.GAME_STARBASE:
cleanupState = StateEvent.GAME_STARBASE_CLEANUP;
break;
case null:
case StateEvent.PRELOAD:
cleanupState = "";
break;
default:
cleanupState = StateEvent.DEFAULT_CLEANUP;
break;
}
transitionEvent.addEvents(new StateEvent(StateEvent.SHUTDOWN_FINISH, cleanupState),
new StateEvent(cleanupState, StateEvent.SHUTDOWN_FINISH));
dispatch(transitionEvent);
}

private function finishShutdown():void
{
dispatch(new ViewEvent(ViewEvent.DESTROY_ALL_VIEWS));
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_COMPLETE);
dispatch(transitionEvent);
}
}
}

-----------------------------------
File 33: igw\com\controller\command\state\StarbaseCoreCommand.as
package com.controller.command.state
{
import com.Application;
import com.event.BattleEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.TransitionEvent;
import com.model.starbase.StarbaseModel;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseCoreCommand extends Command
{
[Inject]
public var event:StarbaseEvent;
[Inject]
public var starbaseModel:StarbaseModel;

override
```

```
public function execute():void
{
switch (event.type)
{
case StarbaseEvent.ENTER_INSTANCED_MISSION:
if(starbaseModel.homeBase.instancedMissionAddress != null)
{
var battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN,
starbaseModel.homeBase.instancedMissionAddress);
dispatch(battleEvent);
}
break;
case StarbaseEvent.ENTER_BASE:
if (event.baseID)
starbaseModel.switchBase(event.baseID);
//ensure that the player's base is not under attack
if (starbaseModel.currentBase.battleServerAddress == null)
{
starbaseModel.entryData = event.viewData;
starbaseModel.entryView = event.view;
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
var cleanupState:String;
switch (Application.STATE)
{
case StateEvent.GAME_BATTLE_INIT:
case StateEvent.GAME_BATTLE:
cleanupState = StateEvent.GAME_BATTLE_CLEANUP;
break;
case StateEvent.GAME_SECTOR_INIT:
case StateEvent.GAME_SECTOR:
cleanupState = StateEvent.GAME_SECTOR_CLEANUP;
break;
case StateEvent.GAME_STARBASE:
cleanupState = StateEvent.GAME_STARBASE_CLEANUP;
break;
case null:
case StateEvent.PRELOAD:
cleanupState = "";
break;
default:
cleanupState = StateEvent.DEFAULT_CLEANUP;
break;
}
transitionEvent.addEvents(new StateEvent(StateEvent.GAME_STARBASE, cleanupState), new
StateEvent(cleanupState, StateEvent.GAME_STARBASE));
dispatch(transitionEvent);
} else
{
//player's base is under attack so go into the battle
var
```

```
battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN,
starbaseModel.currentBase.battleServerAddress);
dispatch(battleEvent);
}
break;
}
}
}
}
```

File 34: igw\com\controller\command\state\StartupCommand.as

```
package com.controller.command.state
{
import com.controller.command.BattleAlertCommand;
import com.controller.command.FTECommand;
import com.controller.command.MissionCommand;
import com.controller.command.PaywallCommand;
import com.controller.command.StarbaseCommand;
import com.controller.command.ToastCommand;
import com.controller.command.TransitionCommand;
import com.controller.command.WelcomeBackCommand;
import com.event.BattleEvent;
import com.event.FTEEvent;
import com.event.MissionEvent;
import com.event.PaywallEvent;
import com.event.ServerEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.ToastEvent;
import com.event.TransitionEvent;
import com.game.entity.factory.IVCFactory;
import com.game.entity.factory.VCFactory;
import com.ui.core.ViewFactory;
import com.ui.core.ViewStack;
import com.ui.core.component.tooltips.Tooltips;

import flash.display.DisplayObjectContainer;

import org.parade.core.IViewFactory;
import org.parade.core.IViewStack;

public class StartupCommand extends StateCommand
{
[Inject]
public var contextView:DisplayObjectContainer;

override public function execute():void
{
mapModel();
mapController();
```

```
mapCommands();
mapView();
startPreloader();
}

private function mapModel():void
{
injector.map(IVCFactory).toSingleton(VCFactory);
}

private function mapView():void
{
injector.map(IViewFactory).toSingleton(ViewFactory);
injector.map(Tooltips).asSingleton();

//create the ViewStack and initialize it
var viewStack:IViewStack = new ViewStack();
injector.injectInto(viewStack);
injector.map(IViewStack).toValue(viewStack);
}

private function mapController():void
{

}

private function mapCommands():void
{
commandMap.map(StateEvent.CREATE_CHARACTER,
null).toCommand(CreateCharacterCommand);
commandMap.map(StateEvent.PRELOAD, StateEvent, true).toCommand(PreloadCommand);
commandMap.map(StateEvent.PRELOAD_COMPLETE, StateEvent,
true).toCommand(PreloadCommand);
commandMap.map(StateEvent.GAME_STARBASE_CLEANUP,
StateEvent).toCommand(GameCommand);
commandMap.map(StateEvent.GAME_BATTLE_CLEANUP,
StateEvent).toCommand(GameCommand);
commandMap.map(StateEvent.GAME_SECTOR_CLEANUP,
StateEvent).toCommand(GameCommand);
commandMap.map(StateEvent.GAME_STARBASE,
StateEvent).toCommand(GameCommand);
commandMap.map(StateEvent.GAME_BATTLE, StateEvent).toCommand(GameCommand);
commandMap.map(StateEvent.GAME_SECTOR, StateEvent).toCommand(GameCommand);
commandMap.map(StateEvent.SHUTDOWN_FINISH,
StateEvent).toCommand(ShutdownCommand);

commandMap.map(FTEEvent.FTE_COMPLETE, FTEEvent).toCommand(FTECommand);
commandMap.map(FTEEvent.FTE_STEP, FTEEvent).toCommand(FTECommand);
commandMap.map(ToastEvent.SHOW_TOAST, ToastEvent).toCommand(ToastCommand);
commandMap.map(TransitionEvent.TRANSITION_BEGIN,
```

```
TransitionEvent).toCommand(TransitionCommand);
commandMap.map(TransitionEvent.TRANSITION_FAILED,
TransitionEvent).toCommand(TransitionCommand);
commandMap.map(TransitionEvent.TRANSITION_COMPLETE,
TransitionEvent).toCommand(TransitionCommand);

commandMap.map(BattleEvent.BATTLE_COUNTDOWN,
BattleEvent).toCommand(BattleAlertCommand);
commandMap.map(BattleEvent.BATTLE_STARTED,
BattleEvent).toCommand(BattleAlertCommand);
commandMap.map(BattleEvent.BATTLE_ENDED,
BattleEvent).toCommand(BattleAlertCommand);
commandMap.map(BattleEvent.BATTLE_REPLAY,
BattleEvent).toCommand(BattleCoreCommand);

commandMap.map(StarbaseEvent.ALERT_FLEET_BATTLE,
StarbaseEvent).toCommand(StarbaseCommand);
commandMap.map(StarbaseEvent.ALERT_STARBASE_BATTLE,
StarbaseEvent).toCommand(StarbaseCommand);
commandMap.map(StarbaseEvent.ALERT_INSTANCED_MISSION_BATTLE,
StarbaseEvent).toCommand(StarbaseCommand);

commandMap.map(StarbaseEvent.WELCOME_BACK,
StarbaseEvent).toCommand(WelcomeBackCommand);

commandMap.map(MissionEvent.MISSION_FAILED,
MissionEvent).toCommand(MissionCommand);
commandMap.map(MissionEvent.MISSION_GREETING,
MissionEvent).toCommand(MissionCommand);
commandMap.map(MissionEvent.MISSION_SITUATIONAL,
MissionEvent).toCommand(MissionCommand);
commandMap.map(MissionEvent.MISSION_VICTORY,
MissionEvent).toCommand(MissionCommand);
commandMap.map(MissionEvent.SHOW_REWARDS,
MissionEvent).toCommand(MissionCommand);

commandMap.map(PaywallEvent.OPEN_PAYWALL,
PaywallEvent).toCommand(PaywallCommand);
}

private function startPreloader():void
{
var connectionEvent:ServerEvent = new ServerEvent(ServerEvent.CONNECT_TO_PROXY);
dispatch(connectionEvent);

var preloaderEvent:StateEvent = new StateEvent(StateEvent.PRELOAD);
dispatch(preloaderEvent);
}

}
}
```

File 35: igw\com\controller\command\state\StartupCoreCommand.as
```
package com.controller.command.state
{
import com.controller.ChatController;
import com.controller.EventController;
import com.controller.GameController;
import com.controller.ServerController;
import com.controller.SettingsController;
import com.controller.command.ContextLostCommand;
import com.controller.command.PaywallCommand;
import com.controller.command.GuestCommand;
import com.controller.command.TransitionCoreCommand;
import com.controller.command.account.ConnectionCommand;
import com.controller.command.account.PaymentCommand;
import com.controller.command.load.LoadCompleteCommand;
import com.controller.command.load.RequestLoadCommand;
import com.controller.fte.FTEController;
import com.controller.keyboard.KeyboardController;
import com.controller.sound.SoundController;
import com.controller.toast.ToastController;
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.IRequirementFactory;
import com.controller.transaction.requirements.RequirementFactory;
import com.event.BattleEvent;
import com.event.LoadEvent;
import com.event.PaywallEvent;
import com.event.RequestLoadEvent;
import com.event.SectorEvent;
import com.event.ServerEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.TransitionEvent;
import com.event.signal.InteractSignal;
import com.event.signal.QuadrantSignal;
import com.game.entity.factory.AttackFactory;
import com.game.entity.factory.BackgroundFactory;
import com.game.entity.factory.IAttackFactory;
import com.game.entity.factory.IBackgroundFactory;
import com.game.entity.factory.IInteractFactory;
import com.game.entity.factory.ISectorFactory;
import com.game.entity.factory.IShipFactory;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.factory.InteractFactory;
import com.game.entity.factory.SectorFactory;
import
```

```
com.game.entity.factory.ShipFactory;
import com.game.entity.factory.StarbaseFactory;
import com.game.entity.factory.VFXFactory;
import com.model.achievements.AchievementModel;
import com.model.alliance.AllianceModel;
import com.model.asset.AssetModel;
import com.model.battle.BattleModel;
import com.model.battlelog.BattleLogModel;
import com.model.blueprint.BlueprintModel;
import com.model.chat.ChatModel;
import com.model.event.EventModel;
import com.model.fleet.FleetModel;
import com.model.leaderboards.LeaderboardModel;
import com.model.mail.MailModel;
import com.model.mission.MissionModel;
import com.model.motd.MotDDailyRewardModel;
import com.model.motd.MotDModel;
import com.model.player.PlayerModel;
import com.model.prototype.PrototypeModel;
import com.model.scene.SceneModel;
import com.model.sector.SectorModel;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.warfrontModel.WarfrontModel;
import com.presenter.battle.BattlePresenter;
import com.presenter.battle.IBattlePresenter;
import com.presenter.battle.IWarfrontPresenter;
import com.presenter.battle.WarfrontPresenter;
import com.presenter.preload.IPreloadPresenter;
import com.presenter.preload.PreloadPresenter;
import com.presenter.sector.IMiniMapPresenter;
import com.presenter.sector.ISectorPresenter;
import com.presenter.sector.MiniMapPresenter;
import com.presenter.sector.SectorPresenter;
import com.presenter.shared.AchievementPresenter;
import com.presenter.shared.AlliancePresenter;
import com.presenter.shared.BookmarkPresenter;
import com.presenter.shared.ChatPresenter;
import com.presenter.shared.CommandPresenter;
import com.presenter.shared.EngineeringPresenter;
import com.presenter.shared.EventPresenter;
import com.presenter.shared.GameOfChancePresenter;
import com.presenter.shared.IAchievementPresenter;
import com.presenter.shared.IAlliancePresenter;
import com.presenter.shared.IBookmarkPresenter;
import com.presenter.shared.IChatPresenter;
import com.presenter.shared.ICommandPresenter;
import com.presenter.shared.IEngineeringPresenter;
import com.presenter.shared.IEventPresenter;
import com.presenter.shared.IGameOfChancePresenter;
import
```

```
com.presenter.shared.ILeaderboardPresenter;
import com.presenter.shared.IPlayerProfilePresenter;
import com.presenter.shared.ITransitionPresenter;
import com.presenter.shared.IUIPresenter;
import com.presenter.shared.LeaderboardPresenter;
import com.presenter.shared.PlayerProfilePresenter;
import com.presenter.shared.TransitionPresenter;
import com.presenter.shared.UIPresenter;
import com.presenter.starbase.AttackAlertPresenter;
import com.presenter.starbase.ConstructionPresenter;
import com.presenter.starbase.FleetPresenter;
import com.presenter.starbase.IAttackAlertPresenter;
import com.presenter.starbase.IConstructionPresenter;
import com.presenter.starbase.IFleetPresenter;
import com.presenter.starbase.IMissionPresenter;
import com.presenter.starbase.IShipyardPresenter;
import com.presenter.starbase.IStarbasePresenter;
import com.presenter.starbase.IStorePresenter;
import com.presenter.starbase.ITradePresenter;
import com.presenter.starbase.MissionPresenter;
import com.presenter.starbase.ShipyardPresenter;
import com.presenter.starbase.StarbasePresenter;
import com.presenter.starbase.StorePresenter;
import com.presenter.starbase.TradePresenter;
import com.service.ExternalInterfaceAPI;
import com.service.kongregate.KongregateAPI;
import com.service.facebook.FacebookAPI;
import com.service.language.Localization;
import com.service.loading.ILoadService;
import com.service.loading.LoadService;
import com.util.AllegianceUtil;
import com.util.BattleUtils;
import com.util.CommonFunctionUtil;
import com.util.RangeBuilder;
import com.util.RouteLineBuilder;
import com.util.statcalc.StatCalcUtil;

import org.as3commons.logging.api.LOGGER_FACTORY;
import org.as3commons.logging.setup.SimpleTargetSetup;
import org.as3commons.logging.setup.target.FlashConsoleTarget;

public class StartupCoreCommand extends StateCommand
{
override public function execute():void
{
//setup the logging and set it to show its' output in the console (Cc)
LOGGER_FACTORY.setup = new SimpleTargetSetup(new FlashConsoleTarget());

mapCommands();
mapControllers();
mapModels();
```

```
mapPresenters();
mapService();
mapSignals();
}

private function mapCommands():void
{
//state events
commandMap.map(StateEvent.GAME_STARBASE_CLEANUP,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_BATTLE_CLEANUP,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_SECTOR_CLEANUP,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.DEFAULT_CLEANUP,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_BATTLE_INIT,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_SECTOR_INIT,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_STARBASE,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_BATTLE,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.GAME_SECTOR,
StateEvent).toCommand(GameCoreCommand);
commandMap.map(StateEvent.SHUTDOWN_START,
StateEvent).toCommand(ShutdownCoreCommand);
commandMap.map(StateEvent.SHUTDOWN_FINISH,
StateEvent).toCommand(ShutdownCoreCommand);

//this command will handle the error that occurs when starling loses the device context
commandMap.map(StateEvent.LOST_CONTEXT, null).toCommand(ContextLostCommand);

commandMap.map(BattleEvent.BATTLE_JOIN,
BattleEvent).toCommand(BattleCoreCommand);

commandMap.map(ServerEvent.CONNECT_TO_PROXY,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.LOGIN_TO_ACCOUNT,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.NEED_CHARACTER_CREATE,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.AUTHORIZED,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.FAILED_TO_CONNECT,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.MAINTENANCE,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.BANNED,
```

```
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.SUSPENSION,
ServerEvent).toCommand(ConnectionCommand);
commandMap.map(ServerEvent.OPEN_PAYMENT,
ServerEvent).toCommand(PaymentCommand);
commandMap.map(ServerEvent.GUEST_RESTRICTION,
ServerEvent).toCommand(GuestCommand);

commandMap.map(SectorEvent.CHANGE_SECTOR,
SectorEvent).toCommand(SectorCoreCommand);
commandMap.map(StarbaseEvent.ENTER_BASE,
StarbaseEvent).toCommand(StarbaseCoreCommand);
commandMap.map(StarbaseEvent.ENTER_INSTANCED_MISSION,
StarbaseEvent).toCommand(StarbaseCoreCommand);

commandMap.map(LoadEvent.COMPLETE,
LoadEvent).toCommand(LoadCompleteCommand);
commandMap.map(RequestLoadEvent.REQUEST,
RequestLoadEvent).toCommand(RequestLoadCommand);
commandMap.map(TransitionEvent.TRANSITION_BEGIN,
TransitionEvent).toCommand(TransitionCoreCommand);
commandMap.map(TransitionEvent.TRANSITION_FAILED,
TransitionEvent).toCommand(TransitionCoreCommand);
commandMap.map(TransitionEvent.TRANSITION_COMPLETE,
TransitionEvent).toCommand(TransitionCoreCommand);


commandMap.map(PaywallEvent.BUY_ITEM, PaywallEvent).toCommand(PaywallCommand);
commandMap.map(PaywallEvent.GET_PAYWALL,
PaywallEvent).toCommand(PaywallCommand);
commandMap.map(PaywallEvent.OPEN_PAYWALL,
PaywallEvent).toCommand(PaywallCommand);
}

private function mapControllers():void
{
injector.map(ChatController).asSingleton();
injector.map(FTEController).asSingleton();
injector.map(GameController).asSingleton();
injector.map(KeyboardController).asSingleton();
injector.map(ServerController).asSingleton();
injector.map(ToastController).asSingleton();
injector.map(TransactionController).asSingleton();
injector.map(SettingsController).asSingleton();
injector.map(SoundController).asSingleton();
injector.map(EventController).asSingleton();
}

private function mapModels():void
{
injector.map(IAttackFactory).toSingleton(AttackFactory);
```

```
injector.map(IBackgroundFactory).toSingleton(BackgroundFactory);
injector.map(IInteractFactory).toSingleton(InteractFactory);
injector.map(IRequirementFactory).toSingleton(RequirementFactory);
injector.map(ISectorFactory).toSingleton(SectorFactory);
injector.map(IShipFactory).toSingleton(ShipFactory);
injector.map(IStarbaseFactory).toSingleton(StarbaseFactory);
injector.map(IVFXFactory).toSingleton(VFXFactory);

injector.map(AllianceModel).asSingleton();
injector.map(AssetModel).asSingleton();
injector.map(BattleLogModel).asSingleton();
injector.map(BattleModel).asSingleton();
injector.map(BlueprintModel).asSingleton();
injector.map(ChatModel).asSingleton();
injector.map(FleetModel).asSingleton();
injector.map(LeaderboardModel).asSingleton();
injector.map(MailModel).asSingleton();
injector.map(MissionModel).asSingleton();
injector.map(MotDDailyRewardModel).asSingleton();
injector.map(MotDModel).asSingleton();
injector.map(PlayerModel).asSingleton();
injector.map(PrototypeModel).asSingleton();
injector.map(SceneModel).asSingleton();
injector.map(SectorModel).asSingleton();
injector.map(StarbaseModel).asSingleton();
injector.map(TransactionModel).asSingleton();
injector.map(WarfrontModel).asSingleton();
injector.map(AchievementModel).asSingleton();
injector.map(EventModel).asSingleton();
}

private function mapPresenters():void
{
injector.map(ITransitionPresenter).toSingleton(TransitionPresenter);
injector.map(IPreloadPresenter).toSingleton(PreloadPresenter);

//game state presenters
injector.map(IStarbasePresenter).toSingleton(StarbasePresenter);
injector.map(IBattlePresenter).toSingleton(BattlePresenter);
injector.map(ISectorPresenter).toSingleton(SectorPresenter);
injector.map(IMiniMapPresenter).toSingleton(MiniMapPresenter);

//ui presenters
injector.map(IAlliancePresenter).toSingleton(AlliancePresenter);
injector.map(IAttackAlertPresenter).toSingleton(AttackAlertPresenter);
injector.map(IConstructionPresenter).toSingleton(ConstructionPresenter);
injector.map(IEngineeringPresenter).toSingleton(EngineeringPresenter);
injector.map(IFleetPresenter).toSingleton(FleetPresenter);
injector.map(IMissionPresenter).toSingleton(MissionPresenter);
injector.map(IPlayerProfilePresenter).toSingleton(PlayerProfilePresenter);
```

```
injector.map(IShipyardPresenter).toSingleton(ShipyardPresenter);
injector.map(IStorePresenter).toSingleton(StorePresenter);
injector.map(ITradePresenter).toSingleton(TradePresenter);
injector.map(IUIPresenter).toSingleton(UIPresenter);
injector.map(IWarfrontPresenter).toSingleton(WarfrontPresenter);
injector.map(ICommandPresenter).toSingleton(CommandPresenter);
injector.map(IChatPresenter).toSingleton(ChatPresenter);
injector.map(IBookmarkPresenter).toSingleton(BookmarkPresenter);
injector.map(IAchievementPresenter).toSingleton(AchievementPresenter);
injector.map(IGameOfChancePresenter).toSingleton(GameOfChancePresenter);
injector.map(ILeaderboardPresenter).toSingleton(LeaderboardPresenter);
injector.map(IEventPresenter).toSingleton(EventPresenter);
}

private function mapService():void
{
injector.map(AllegianceUtil).toSingleton(AllegianceUtil);
injector.map(KongregateAPI).toSingleton(KongregateAPI);
injector.map(FacebookAPI).toSingleton(FacebookAPI);
injector.map(ILoadService).toSingleton(LoadService);
injector.map(RangeBuilder).asSingleton();
injector.map(RouteLineBuilder).asSingleton();

injector.injectInto(new BattleUtils());
injector.injectInto(new CommonFunctionUtil());
injector.injectInto(new ExternalInterfaceAPI());
injector.injectInto(new Localization());
injector.injectInto(new StatCalcUtil());
}

private function mapSignals():void
{
var ints:InteractSignal = new InteractSignal();
var qs:QuadrantSignal = new QuadrantSignal();

injector.map(InteractSignal).toValue(ints);
injector.map(QuadrantSignal).toValue(qs);
}
}
}



---
File 36: igw\com\controller\command\state\StateCommand.as
package com.controller.command.state
{
import com.event.StateEvent;

import
```

```
org.robotlegs.extensions.presenter.impl.Command;

public class StateCommand extends Command
{
[Inject]
public var event:StateEvent;
}
}
-----------------------------------
File 37: igw\com\controller\fte\FTEController.as
package com.controller.fte
{
import com.Application;
import com.controller.ServerController;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.FTEEvent;
import com.event.SectorEvent;
import com.model.mission.MissionModel;
import com.model.mission.MissionVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.outgoing.proxy.ProxyTutorialStepCompletedMessage;
import com.service.server.outgoing.starbase.StarbaseSkipTrainingRequest;
import com.ui.core.View;
import com.ui.core.component.contextmenu.ContextMenu;

import flash.events.Event;

import flash.events.IEventDispatcher;
import flash.events.TimerEvent;
import flash.utils.Timer;
import flash.utils.getDefinitionByName;

import org.parade.core.ViewController;
import org.parade.enum.PlatformEnum;
import org.parade.enum.ViewEnum;

import com.service.ExternalInterfaceAPI;

public class FTEController
{
private var _complete:Boolean = false;
private var _stepVO:FTEStepVO = new FTEStepVO();
private var _eventDispatcher:IEventDispatcher;
private var _missionModel:MissionModel;
private var _progressStepOnStateChange:Boolean = false;
private var _prototypeModel:PrototypeModel;
private var _ready:Boolean = false;
private
```

```
var _step:int = 0;
private var _timeDelay:Timer;
private var _viewController:ViewController;

public var closeContext:ContextMenu;

public var serverController:ServerController;

[PostConstruct]
public function init():void
{
_timeDelay = new Timer(0, 1);
_timeDelay.addEventListener(TimerEvent.TIMER_COMPLETE, dispatch, false, 0, true);
}

public function skipFTE():void
{
if (serverController)
{
ExternalInterfaceAPI.logConsole("Skip Tutorial");
var msg:ProxyTutorialStepCompletedMessage = ProxyTutorialStepCompletedMessage(
serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_TUTORIAL_STEP_COMPLETED));
msg.stepId = 411060;
msg.kabamNaid = CurrentUser.naid;
serverController.send(msg);

var skipTrainingRequest:StarbaseSkipTrainingRequest =
StarbaseSkipTrainingRequest(serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_SKIP_TRAINING_MESSAGE));
serverController.send(skipTrainingRequest);

var event:Event;
event = new SectorEvent(SectorEvent.CHANGE_SECTOR, null);
_eventDispatcher.dispatchEvent(event);
}
}

public function nextStep():void
{
if (!_complete)
{
if (serverController && _stepVO.stepId != 0)
{
var msg:ProxyTutorialStepCompletedMessage = ProxyTutorialStepCompletedMessage(
serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_TUTORIAL_STEP_COMPLETED));
msg.stepId = _stepVO.stepId;
msg.kabamNaid = CurrentUser.naid;
serverController.send(msg);
}
```

```
_progressStepOnStateChange = false;
_step++;

if (_timeDelay)
_timeDelay.reset();

if (closeContext)
{
closeContext.destroy();
closeContext = null;
}

if (!_complete && _step >= _stepVO.totalSteps)
{
_complete = true;
showNextStep();
cleanup();
} else
{
_stepVO.updateStep(_step);
showNextStep();
}
}
}

public function checkMissionRequired( mission:MissionVO ):void
{
var completedCurrentStep:Boolean = false;
if (!_stepVO || !_stepVO.step)
return;
if (_stepVO.missionName.length > 0)
{
var names:Array = _stepVO.missionName.split(',');
var index:int = names.indexOf(mission.name);
if (index > -1)
completedCurrentStep = true;
}
if (completedCurrentStep)
nextStep();
else if (_stepVO && !_stepVO.anchor)
{
//determine which step the player is on based on the mission
var missionName:String;
var steps:Vector.<IPrototype> =
_prototypeModel.getFTEStepsByPlatform(PlatformEnum.BROWSER); //hardcoding this to
browser for now
for (var i:int = 0; i < steps.length; i++)
{
missionName
```

```
= steps[i].getValue("missionName");
if (missionName.length > 0)
{
names = missionName.split(',');
if (names.indexOf(mission.name) > -1)
{
if (_stepVO.currentStep < i + 1)
{
//remove any old view we may be looking at
var c:Class;
var newViewClass:Class = (steps[i + 1].getValue('uiID') != "") ?
Class(getDefinitionByName(steps[i + 1].getValue('uiID'))) : null;
var view:View;
if (_stepVO.currentStep != 0)
{
//go back through the steps to find the last view
for (var j:int = i - 1; j >= 0; j--)
{
if (steps[j].getValue('uiID') != "")
{
c = Class(getDefinitionByName(steps[j].getValue('uiID')));
if (c != newViewClass)
{
view = View(_viewController.getView(c));
if (view && view.type != ViewEnum.UI)
{
if (view.parent)
view.destroy();
else
_viewController.removeFromQueue(_stepVO.viewClass);
}
}
break;
}
}
}

//start on the step right after
_step = i + 1;
_stepVO.updateStep(_step);
showNextStep();
}
break;
}
}
}
}
}

private function showNextStep():void
{
```

```
if (_complete || _stepVO.timeDelay <= 0)
dispatch();
else
{
_timeDelay.delay = _stepVO.timeDelay;
_timeDelay.start();
}
}

public function inFTE():Boolean { return _missionModel.currentMission.isFTE; }

private function checkStart():void
{
if (_ready)
{
//look at which mission the player is on to determine if they are in the fte
var mission:MissionVO = _missionModel.currentMission;
if (mission.isFTE)
{
//setup the steps
var steps:Vector.<IPrototype> =
_prototypeModel.getFTEStepsByPlatform(PlatformEnum.BROWSER); //hardcoding this to
browser for now
_stepVO = new FTEStepVO();
_stepVO.init(steps);
_stepVO.updateStep(_step);

// Block Flash's core accessibility (tab to select) feature
Application.STAGE.tabChildren = false;

checkMissionRequired(mission);
if (_stepVO.currentStep == 0)
showNextStep();
} else
{
//if we can't find a match to the player's current mission, then the player must be done with the
fte
_complete = true;
cleanup();
}
}
}

private function dispatch( e:TimerEvent = null ):void
{
var event:FTEEvent;
if (_complete)
event = new FTEEvent(FTEEvent.FTE_COMPLETE, _stepVO);
else
event
```

```
= new FTEEvent(FTEEvent.FTE_STEP, _stepVO);
_eventDispatcher.dispatchEvent(event);
}

public function get complete():Boolean { return _complete; }
public function get progressStepOnStateChange():Boolean { return
_progressStepOnStateChange; }
public function set progressStepOnStateChange( v:Boolean ):void {
_progressStepOnStateChange = v; }
public function get step():int { return _step; }

[Inject]
public function set eventDispatcher( v:IEventDispatcher ):void { _eventDispatcher = v; }
[Inject]
public function set missionModel( v:MissionModel ):void { _missionModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set viewController( v:ViewController ):void { _viewController = v; }

public function set ready( v:Boolean ):void
{
if (!_ready && v)
{
_ready = true;
checkStart();
}
}
public function get running():Boolean { return (_ready && !_complete) || (_missionModel &&
_missionModel.currentMission && _missionModel.currentMission.isFTE); }
public function get startInSector():Boolean
{
if (_complete)
return false;

var mission:MissionVO = _missionModel.currentMission;
switch (mission.name)
{
case "FTE_IGA_Intro":
case "FTE_SOV_Intro":
case "FTE_TYR_Intro":

case "FTE_IGA_Starting_Mission":
case "FTE_SOV_Starting_Mission":
case "FTE_TYR_Starting_Mission":

case "FTE_IGA_Dock":
case "FTE_SOV_Dock":
case "FTE_TYR_Dock":

case
```

```
"FTE_TYR_Upgrade_Shipyard_Begin":
case "FTE_IGA_Upgrade_Shipyard_Begin":
case "FTE_SOV_Upgrade_Shipyard_Begin":

case "FTE_TYR_Reward":
case "FTE_IGA_Reward":
case "FTE_SOV_Reward":

return true;
break;
}
return false;
}

public function terminate():void
{
_complete = true;
dispatch();
}

private function cleanup():void
{
_eventDispatcher = null;
_missionModel = null;
_prototypeModel = null;
_stepVO = null;
if (_timeDelay.running)
_timeDelay.stop();
_timeDelay = null;
_viewController = null;

// Turn accessibility back on
Application.STAGE.tabChildren = true;
}
}
}



---------------------------------
File 38: igw\com\controller\fte\FTEStepVO.as
package com.controller.fte
{
import com.model.prototype.IPrototype;
import com.service.language.Localization;

import flash.geom.Point;
import flash.geom.Rectangle;
import flash.utils.getDefinitionByName;

public
```

```
class FTEStepVO
{
private var _arrowPosition:Point;
private var _arrowRotation:Number = -1;
private var _currentStep:int = 0;
private var _cutout:Rectangle;
private var _step:IPrototype;
private var _steps:Vector.<IPrototype>;

public function init( steps:Vector.<IPrototype> ):void { _steps = steps; }

public function updateStep( currentStep:int ):void
{
if (currentStep < _steps.length)
{
_currentStep = currentStep;
_step = _steps[currentStep];

var coords:Array;
//set the cutout
if (_step.getValue('cutoutCoordinates') != '' && _step.getValue('cutoutCoordinates') != null)
{
coords = String(_step.getValue('cutoutCoordinates')).split(',');
_cutout = new Rectangle(coords[0], coords[1], coords[2], coords[3]);
} else
_cutout = null;

//set the arrow position
if (_step.getValue('arrowCoordinates') != '' && _step.getValue('arrowCoordinates') != null)
{
coords = String(_step.getValue('arrowCoordinates')).split(',');
_arrowPosition = new Point(coords[0], coords[1]);
_arrowRotation = coords[2];
} else
_arrowPosition = null;
}
}

public function unescape( str:String ):String
{
return str.replace(/\\n/g, '\n');
}

public function get anchor():Boolean { return _step.getValue('anchor'); }
public function get arrowPosition():Point { return _arrowPosition; }
public function set arrowPosition( v:Point ):void { _arrowPosition = v; }
public function get arrowRotation():Number { return _arrowRotation; }
public function set arrowRotation( v:Number ):void { _arrowRotation = v; }
public function get currentStep():int { return _currentStep; }
public function get cutout():Rectangle { return _cutout; }
public
```

```
function set cutout( v:Rectangle ):void { _cutout = v; }
public function get dialog():String { return
unescape(Localization.instance.getString(_step.getValue('dialogString'))); }
public function get audioDir():String { return _step.getValue('dialogAudioString'); }
public function get titleText():String { return
Localization.instance.getString(_step.getValue('dialogString') + '.Title'); }
public function get missionName():String { return _step.getValue('missionName'); }
public function get mood():int { return _step.getValue('mood'); }
public function get platform():String { return _step.getValue('platform'); }
public function get step():IPrototype { return _step; }
public function get stepId():int { return _step.getValue("stepId"); }
public function get timeDelay():int { return _step.getValue('timeDelay'); }
public function get totalSteps():int { return _steps.length; }
public function get trigger():String
{
if (_step.getValue('trigger') == "" || _step.getValue('trigger') == null)
return null;
return _step.getValue('trigger');
}

public function get viewClass():Class
{
if (_step.getValue('uiID') != "" && _step.getValue('uiID') != null)
{
try
{
return Class(getDefinitionByName(_step.getValue('uiID')));
} catch ( e:Error )
{
//just do nothing
}
}
return null;
}
public function get voiceOver():String { return _step.getValue('voID'); }

public function get name():String { return _step.getUnsafeValue('name'); }
}
}

-----------------------------------
File 39: igw\com\controller\keyboard\KeyboardController.as
package com.controller.keyboard
{
import flash.display.Stage;
import flash.events.KeyboardEvent;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;

public
```

```
class KeyboardController
{
[Inject]
public var stage:Stage;

protected var _keyStateDown:Dictionary = new Dictionary();
protected var _keyStateUp:Dictionary = new Dictionary();

[PostConstruct]
public function init():void
{
stage.addEventListener(KeyboardEvent.KEY_DOWN, onKeyDown);
stage.addEventListener(KeyboardEvent.KEY_UP, onKeyUp);
}

public function addKeyDownListener( callback:Function, keyCode:uint ):void
{
if (!_keyStateDown[keyCode])
_keyStateDown[keyCode] = new Signal(uint);
_keyStateDown[keyCode].add(callback);
}

public function removeKeyDownListener( callback:Function, keyCode:uint ):void
{
if (!_keyStateDown.hasOwnProperty(keyCode))
return;
_keyStateDown[keyCode].remove(callback);
}

public function addKeyUpListener( callback:Function, keyCode:uint ):void
{
if (!_keyStateUp[keyCode])
_keyStateUp[keyCode] = new Signal(uint);
_keyStateUp[keyCode].add(callback);
}

public function removeKeyUpListener( callback:Function, keyCode:uint ):void
{
if (!_keyStateUp.hasOwnProperty(keyCode))
return;
_keyStateUp[keyCode].remove(callback);
}

protected function onKeyDown( ke:KeyboardEvent ):void
{
if (!_keyStateDown.hasOwnProperty(ke.keyCode))
return;
if (_keyStateDown[ke.keyCode].numListeners > 0)
_keyStateDown[ke.keyCode].dispatch(ke.keyCode);
}

protected
```

```
function onKeyUp( ke:KeyboardEvent ):void
{
if (!_keyStateUp.hasOwnProperty(ke.keyCode))
return;
if (_keyStateUp[ke.keyCode].numListeners > 0)
_keyStateUp[ke.keyCode].dispatch(ke.keyCode);
}

public function destroy():void
{
stage.removeEventListener(KeyboardEvent.KEY_DOWN, onKeyDown);
stage.removeEventListener(KeyboardEvent.KEY_UP, onKeyUp);
}
}
}
```

File 40: igw\com\controller\keyboard\KeyboardKey.as

```
/*******************************************************************************
* Smash Engine
* Copyright (C) 2009 Smash Labs, LLC
* For more information see http://www.Smashengine.com
*
* This file is licensed under the terms of the MIT license, which is included
* in the License.html file at the root directory of this SDK.
*******************************************************************************/
package com.controller.keyboard
{
import flash.utils.Dictionary;

/**
* Enumeration class that maps friendly key names to their key code equivalent. This class
* should not be instantiated directly, rather, one of the constants should be used.
*/
public class KeyboardKey
{
public static const INVALID:KeyboardKey = new KeyboardKey(0);

public static const BACKSPACE:KeyboardKey = new KeyboardKey(8);
public static const TAB:KeyboardKey = new KeyboardKey(9);
public static const ENTER:KeyboardKey = new KeyboardKey(13);
public static const COMMAND:KeyboardKey = new KeyboardKey(15);
public static const SHIFT:KeyboardKey = new KeyboardKey(16);
public static const CONTROL:KeyboardKey = new KeyboardKey(17);
public static const ALT:KeyboardKey = new KeyboardKey(18);
public static const PAUSE:KeyboardKey = new KeyboardKey(19);
public static const CAPS_LOCK:KeyboardKey = new KeyboardKey(20);
public static const ESCAPE:KeyboardKey = new KeyboardKey(27);

public static const SPACE:KeyboardKey = new KeyboardKey(32);
public
```

```
static const PAGE_UP:KeyboardKey = new KeyboardKey(33);
public static const PAGE_DOWN:KeyboardKey = new KeyboardKey(34);
public static const END:KeyboardKey = new KeyboardKey(35);
public static const HOME:KeyboardKey = new KeyboardKey(36);
public static const LEFT:KeyboardKey = new KeyboardKey(37);
public static const UP:KeyboardKey = new KeyboardKey(38);
public static const RIGHT:KeyboardKey = new KeyboardKey(39);
public static const DOWN:KeyboardKey = new KeyboardKey(40);

public static const INSERT:KeyboardKey = new KeyboardKey(45);
public static const DELETE:KeyboardKey = new KeyboardKey(46);

public static const ZERO:KeyboardKey = new KeyboardKey(48);
public static const ONE:KeyboardKey = new KeyboardKey(49);
public static const TWO:KeyboardKey = new KeyboardKey(50);
public static const THREE:KeyboardKey = new KeyboardKey(51);
public static const FOUR:KeyboardKey = new KeyboardKey(52);
public static const FIVE:KeyboardKey = new KeyboardKey(53);
public static const SIX:KeyboardKey = new KeyboardKey(54);
public static const SEVEN:KeyboardKey = new KeyboardKey(55);
public static const EIGHT:KeyboardKey = new KeyboardKey(56);
public static const NINE:KeyboardKey = new KeyboardKey(57);

public static const A:KeyboardKey = new KeyboardKey(65);
public static const B:KeyboardKey = new KeyboardKey(66);
public static const C:KeyboardKey = new KeyboardKey(67);
public static const D:KeyboardKey = new KeyboardKey(68);
public static const E:KeyboardKey = new KeyboardKey(69);
public static const F:KeyboardKey = new KeyboardKey(70);
public static const G:KeyboardKey = new KeyboardKey(71);
public static const H:KeyboardKey = new KeyboardKey(72);
public static const I:KeyboardKey = new KeyboardKey(73);
public static const J:KeyboardKey = new KeyboardKey(74);
public static const K:KeyboardKey = new KeyboardKey(75);
public static const L:KeyboardKey = new KeyboardKey(76);
public static const M:KeyboardKey = new KeyboardKey(77);
public static const N:KeyboardKey = new KeyboardKey(78);
public static const O:KeyboardKey = new KeyboardKey(79);
public static const P:KeyboardKey = new KeyboardKey(80);
public static const Q:KeyboardKey = new KeyboardKey(81);
public static const R:KeyboardKey = new KeyboardKey(82);
public static const S:KeyboardKey = new KeyboardKey(83);
public static const T:KeyboardKey = new KeyboardKey(84);
public static const U:KeyboardKey = new KeyboardKey(85);
public static const V:KeyboardKey = new KeyboardKey(86);
public static const W:KeyboardKey = new KeyboardKey(87);
public static const X:KeyboardKey = new KeyboardKey(88);
public static const Y:KeyboardKey = new KeyboardKey(89);
public static const Z:KeyboardKey = new KeyboardKey(90);

public
```

```
static const NUM0:KeyboardKey = new KeyboardKey(96);
public static const NUM1:KeyboardKey = new KeyboardKey(97);
public static const NUM2:KeyboardKey = new KeyboardKey(98);
public static const NUM3:KeyboardKey = new KeyboardKey(99);
public static const NUM4:KeyboardKey = new KeyboardKey(100);
public static const NUM5:KeyboardKey = new KeyboardKey(101);
public static const NUM6:KeyboardKey = new KeyboardKey(102);
public static const NUM7:KeyboardKey = new KeyboardKey(103);
public static const NUM8:KeyboardKey = new KeyboardKey(104);
public static const NUM9:KeyboardKey = new KeyboardKey(105);

public static const MULTIPLY:KeyboardKey = new KeyboardKey(106);
public static const ADD:KeyboardKey = new KeyboardKey(107);
public static const NUMENTER:KeyboardKey = new KeyboardKey(108);
public static const SUBTRACT:KeyboardKey = new KeyboardKey(109);
public static const DECIMAL:KeyboardKey = new KeyboardKey(110);
public static const DIVIDE:KeyboardKey = new KeyboardKey(111);

public static const F1:KeyboardKey = new KeyboardKey(112);
public static const F2:KeyboardKey = new KeyboardKey(113);
public static const F3:KeyboardKey = new KeyboardKey(114);
public static const F4:KeyboardKey = new KeyboardKey(115);
public static const F5:KeyboardKey = new KeyboardKey(116);
public static const F6:KeyboardKey = new KeyboardKey(117);
public static const F7:KeyboardKey = new KeyboardKey(118);
public static const F8:KeyboardKey = new KeyboardKey(119);
public static const F9:KeyboardKey = new KeyboardKey(120);
// F10 is considered 'reserved' by Flash
public static const F11:KeyboardKey = new KeyboardKey(122);
public static const F12:KeyboardKey = new KeyboardKey(123);

public static const NUM_LOCK:KeyboardKey = new KeyboardKey(144);
public static const SCROLL_LOCK:KeyboardKey = new KeyboardKey(145);

public static const COLON:KeyboardKey = new KeyboardKey(186);
public static const PLUS:KeyboardKey = new KeyboardKey(187);
public static const COMMA:KeyboardKey = new KeyboardKey(188);
public static const MINUS:KeyboardKey = new KeyboardKey(189);
public static const PERIOD:KeyboardKey = new KeyboardKey(190);
public static const BACKSLASH:KeyboardKey = new KeyboardKey(191);
public static const TILDE:KeyboardKey = new KeyboardKey(192);

public static const LEFT_BRACKET:KeyboardKey = new KeyboardKey(219);
public static const SLASH:KeyboardKey = new KeyboardKey(220);
public static const RIGHT_BRACKET:KeyboardKey = new KeyboardKey(221);
public static const QUOTE:KeyboardKey = new KeyboardKey(222);

/**
* A dictionary mapping the string names of all the keys to the InputKey they represent.
*/
public
```

```
static function get staticTypeMap():Dictionary
{
if (!_typeMap)
{
_typeMap = new Dictionary();
_typeMap["BACKSPACE"] = BACKSPACE;
_typeMap["TAB"] = TAB;
_typeMap["ENTER"] = ENTER;
_typeMap["RETURN"] = ENTER;
_typeMap["SHIFT"] = SHIFT;
_typeMap["COMMAND"] = COMMAND;
_typeMap["CONTROL"] = CONTROL;
_typeMap["ALT"] = ALT;
_typeMap["OPTION"] = ALT;
_typeMap["ALTERNATE"] = ALT;
_typeMap["PAUSE"] = PAUSE;
_typeMap["CAPS_LOCK"] = CAPS_LOCK;
_typeMap["ESCAPE"] = ESCAPE;
_typeMap["SPACE"] = SPACE;
_typeMap["SPACE_BAR"] = SPACE;
_typeMap["PAGE_UP"] = PAGE_UP;
_typeMap["PAGE_DOWN"] = PAGE_DOWN;
_typeMap["END"] = END;
_typeMap["HOME"] = HOME;
_typeMap["LEFT"] = LEFT;
_typeMap["UP"] = UP;
_typeMap["RIGHT"] = RIGHT;
_typeMap["DOWN"] = DOWN;
_typeMap["LEFT_ARROW"] = LEFT;
_typeMap["UP_ARROW"] = UP;
_typeMap["RIGHT_ARROW"] = RIGHT;
_typeMap["DOWN_ARROW"] = DOWN;
_typeMap["INSERT"] = INSERT;
_typeMap["DELETE"] = DELETE;
_typeMap["ZERO"] = ZERO;
_typeMap["ONE"] = ONE;
_typeMap["TWO"] = TWO;
_typeMap["THREE"] = THREE;
_typeMap["FOUR"] = FOUR;
_typeMap["FIVE"] = FIVE;
_typeMap["SIX"] = SIX;
_typeMap["SEVEN"] = SEVEN;
_typeMap["EIGHT"] = EIGHT;
_typeMap["NINE"] = NINE;
_typeMap["0"] = ZERO;
_typeMap["1"] = ONE;
_typeMap["2"] = TWO;
_typeMap["3"] = THREE;
_typeMap["4"] = FOUR;
_typeMap["5"] = FIVE;
_typeMap["6"]
```

```
= SIX;
_typeMap["7"] = SEVEN;
_typeMap["8"] = EIGHT;
_typeMap["9"] = NINE;
_typeMap["NUMBER_0"] = ZERO;
_typeMap["NUMBER_1"] = ONE;
_typeMap["NUMBER_2"] = TWO;
_typeMap["NUMBER_3"] = THREE;
_typeMap["NUMBER_4"] = FOUR;
_typeMap["NUMBER_5"] = FIVE;
_typeMap["NUMBER_6"] = SIX;
_typeMap["NUMBER_7"] = SEVEN;
_typeMap["NUMBER_8"] = EIGHT;
_typeMap["NUMBER_9"] = NINE;
_typeMap["A"] = A;
_typeMap["B"] = B;
_typeMap["C"] = C;
_typeMap["D"] = D;
_typeMap["E"] = E;
_typeMap["F"] = F;
_typeMap["G"] = G;
_typeMap["H"] = H;
_typeMap["I"] = I;
_typeMap["J"] = J;
_typeMap["K"] = K;
_typeMap["L"] = L;
_typeMap["M"] = M;
_typeMap["N"] = N;
_typeMap["O"] = O;
_typeMap["P"] = P;
_typeMap["Q"] = Q;
_typeMap["R"] = R;
_typeMap["S"] = S;
_typeMap["T"] = T;
_typeMap["U"] = U;
_typeMap["V"] = V;
_typeMap["W"] = W;
_typeMap["X"] = X;
_typeMap["Y"] = Y;
_typeMap["Z"] = Z;
_typeMap["NUM0"] = NUM0;
_typeMap["NUM1"] = NUM1;
_typeMap["NUM2"] = NUM2;
_typeMap["NUM3"] = NUM3;
_typeMap["NUM4"] = NUM4;
_typeMap["NUM5"] = NUM5;
_typeMap["NUM6"] = NUM6;
_typeMap["NUM7"] = NUM7;
_typeMap["NUM8"] = NUM8;
_typeMap["NUM9"] = NUM9;
_typeMap["NUMPAD_0"]
```

```
= NUM0;
_typeMap["NUMPAD_1"] = NUM1;
_typeMap["NUMPAD_2"] = NUM2;
_typeMap["NUMPAD_3"] = NUM3;
_typeMap["NUMPAD_4"] = NUM4;
_typeMap["NUMPAD_5"] = NUM5;
_typeMap["NUMPAD_6"] = NUM6;
_typeMap["NUMPAD_7"] = NUM7;
_typeMap["NUMPAD_8"] = NUM8;
_typeMap["NUMPAD_9"] = NUM9;
_typeMap["MULTIPLY"] = MULTIPLY;
_typeMap["ASTERISK"] = MULTIPLY;
_typeMap["NUMMULTIPLY"] = MULTIPLY;
_typeMap["NUMPAD_MULTIPLY"] = MULTIPLY;
_typeMap["ADD"] = ADD;
_typeMap["NUMADD"] = ADD;
_typeMap["NUMPAD_ADD"] = ADD;
_typeMap["SUBTRACT"] = SUBTRACT;
_typeMap["NUMSUBTRACT"] = SUBTRACT;
_typeMap["NUMPAD_SUBTRACT"] = SUBTRACT;
_typeMap["DECIMAL"] = DECIMAL;
_typeMap["NUMDECIMAL"] = DECIMAL;
_typeMap["NUMPAD_DECIMAL"] = DECIMAL;
_typeMap["DIVIDE"] = DIVIDE;
_typeMap["NUMDIVIDE"] = DIVIDE;
_typeMap["NUMPAD_DIVIDE"] = DIVIDE;
_typeMap["NUMENTER"] = NUMENTER;
_typeMap["NUMPAD_ENTER"] = NUMENTER;
_typeMap["F1"] = F1;
_typeMap["F2"] = F2;
_typeMap["F3"] = F3;
_typeMap["F4"] = F4;
_typeMap["F5"] = F5;
_typeMap["F6"] = F6;
_typeMap["F7"] = F7;
_typeMap["F8"] = F8;
_typeMap["F9"] = F9;
_typeMap["F11"] = F11;
_typeMap["F12"] = F12;
_typeMap["NUM_LOCK"] = NUM_LOCK;
_typeMap["SCROLL_LOCK"] = SCROLL_LOCK;
_typeMap["COLON"] = COLON;
_typeMap["SEMICOLON"] = COLON;
_typeMap["PLUS"] = PLUS;
_typeMap["EQUAL"] = PLUS;
_typeMap["COMMA"] = COMMA;
_typeMap["LESS_THAN"] = COMMA;
_typeMap["MINUS"] = MINUS;
_typeMap["UNDERSCORE"] = MINUS;
_typeMap["PERIOD"] = PERIOD;
_typeMap["GREATER_THAN"]
```

```
= PERIOD;
_typeMap["BACKSLASH"] = BACKSLASH;
_typeMap["QUESTION_MARK"] = BACKSLASH;
_typeMap["TILDE"] = TILDE;
_typeMap["BACK_QUOTE"] = TILDE;
_typeMap["LEFT_BRACKET"] = LEFT_BRACKET;
_typeMap["LEFT_BRACE"] = LEFT_BRACKET;
_typeMap["SLASH"] = SLASH;
_typeMap["FORWARD_SLASH"] = SLASH;
_typeMap["PIPE"] = SLASH;
_typeMap["RIGHT_BRACKET"] = RIGHT_BRACKET;
_typeMap["RIGHT_BRACE"] = RIGHT_BRACKET;
_typeMap["QUOTE"] = QUOTE;
}

return _typeMap;
}

/**
* Converts a key code to the string that represents it.
*/
public static function codeToString( value:int ):String
{
var tm:Dictionary = staticTypeMap;
for (var name:String in tm)
{
if (staticTypeMap[name.toUpperCase()].keyCode == value)
return name.toUpperCase();
}

return null;
}

/**
* Converts the name of a key to the keycode it represents.
*/
public static function stringToCode( value:String ):int
{
if (!staticTypeMap[value.toUpperCase()])
return 0;

return staticTypeMap[value.toUpperCase()].keyCode;
}

/**
* Converts the name of a key to the InputKey it represents.
*/
public static function stringToKey( value:String ):KeyboardKey
{
return staticTypeMap[value.toUpperCase()];
}
```

```
private static var _typeMap:Dictionary = null;

/**
* The key code that this wraps.
*/
public function get keyCode():int
{
return _keyCode;
}

public function KeyboardKey( keyCode:int = 0 )
{
_keyCode = keyCode;
}

public function get typeMap():Dictionary
{
return staticTypeMap;
}

private var _keyCode:int = 0;
}
}

------------------------------------
File 41: igw\com\controller\sound\SoundController.as
package com.controller.sound
{
import com.enum.AudioEnum;
import com.model.asset.AssetModel;
import com.service.loading.LoadPriority;

import flash.events.EventDispatcher;
import flash.media.Sound;
import flash.utils.Dictionary;
import flash.utils.getQualifiedClassName;

import org.greensock.plugins.TweenPlugin;
import org.greensock.plugins.VolumePlugin;
import org.shared.ObjectPool;

public class SoundController extends EventDispatcher
{
//- PRIVATE & PROTECTED VARIABLES
------------------------------------------------------------------------
private static const HIGHLIGHT_SCALAR:Number = .2;

// singleton instance
public
```

```
static var instance:SoundController;

private var _areSFXMuted:Boolean = false;
private var _sfxVolume:Number = 1.0;
private var _assetModel:AssetModel;
private var _currentVoiceOver:String;
private var _currentMusic:String;
private var _isMusicMuted:Boolean = false;
private var _musicVolume:Number = 1.0;
private var _sounds:Vector.<SoundItem>;
private var _soundsRef:Dictionary;
private var _volumeScalar:Number = 1;

public function SoundController()
{
TweenPlugin.activate([VolumePlugin]);
instance = this;
_sounds = new Vector.<SoundItem>;
_soundsRef = new Dictionary(true);

SoundItem.FADE_COMPLETE_SIGNAL.add(handleFadeComplete);
SoundItem.PLAY_COMPLETE_SIGNAL.add(handleSoundPlayComplete);
}

public function addAudio( urlToLoad:String = null, preloadedSound:Sound = null ):void
{
var si:SoundItem;
if (preloadedSound)
{
//a sound has been loaded in so lets add it to the sound item
si = _soundsRef[urlToLoad];
if (!si)
si = createNewSoundItem(urlToLoad);
si.sound = preloadedSound;
si.loading = false;
si.isLoaded = true;

playSound(urlToLoad, si.savedVolume, si.startTime, si.loops);

//clear the cache from the asset model so that the sound can be reloaded at a later time if
needed
_assetModel.removeFromCache(urlToLoad);
} else if (urlToLoad)
{
//make sure a soundItem was created for this sound before loading it
if (_soundsRef.hasOwnProperty(urlToLoad))
{
si = _soundsRef[urlToLoad];
//make sure the type is not muted before we laod it
if (((si.type == AudioEnum.TYPE_SFX || si.type == AudioEnum.TYPE_VOICEOVER) &&
!_areSFXMuted)
```

```
||
(si.type == AudioEnum.TYPE_MUSIC && !_isMusicMuted))
{
si.loading = true;
_assetModel.getFromCache(urlToLoad, null, LoadPriority.DONT_GIVE_A_SHIT);
} else if (si.type == AudioEnum.TYPE_MUSIC) //save out to _currentMusic so when unmuted we can play it
_currentMusic = si.name;
}
}
}

public function removeAudio( name:String ):void
{
if (_soundsRef[name])
{
_soundsRef[name] = null;
delete _soundsRef[name];

var len:int = _sounds.length;
for (var i:int = 0; i < len; i++)
{
if (_sounds[i].name == name)
{
ObjectPool.give(_sounds[i]);
_sounds.splice(i, 1);
break;
}
}
}
}

public function removeAllAudio():void
{
_sounds.length = 0;
_soundsRef = new Dictionary(true);
}

/**
* Plays or resumes a sound from the sound dictionary with the specified name. If the sounds in the dictionary were muted by
* the muteAllSounds() method, no sounds are played until unmuteAllSounds() is called.
*
* @param $name The string identifier of the sound to play
* @param $volume A number from 0 to 1 representing the volume at which to play the sound (default: 1)
* @param $startTime A number (in milliseconds) representing the time to start playing the sound at (default: 0)
* @param $loops An integer representing the number of times to loop the sound (default: 0)
* @param $resumeTween A boolean that indicates if a faded sound's volume should resume from
```

```
the last saved state (default: true)
*
* @return void
*/
public function playSound( name:String, volume:Number = 1, startTime:Number = 0, loops:int =
0, resumeTween:Boolean = true ):void
{
if (_soundsRef[name])
{
var si:SoundItem = SoundItem(_soundsRef[name]);
if (!si.sound)
{
//see if we need to laod this due to being muted before
if (!si.loading && ((si.type == AudioEnum.TYPE_SFX || si.type ==
AudioEnum.TYPE_VOICEOVER) && !_areSFXMuted) ||
(si.type == AudioEnum.TYPE_MUSIC && !_isMusicMuted))
{
addAudio(si.name);
} else if (si.type == AudioEnum.TYPE_MUSIC)
_currentMusic = si.name;
return;
}

if (si.type == AudioEnum.TYPE_VOICEOVER)
{
if (_areSFXMuted)
return;
//stop the old voiceover and remove it from memory
if (_currentVoiceOver)
stopSound(_currentVoiceOver, _currentVoiceOver != name);
_currentVoiceOver = name;

si.play(startTime, loops, volume * _sfxVolume, resumeTween);

//reduce the volume of all other sounds while the voiceover is playing
highlightSound(name, HIGHLIGHT_SCALAR);
} else if (si.type == AudioEnum.TYPE_MUSIC)
{
if (_isMusicMuted)
return;
//stop the old music and remove it from memory
if (_currentMusic && _currentMusic != name)
stopSound(_currentMusic);
si.play(startTime, loops, volume * _musicVolume, resumeTween);
_currentMusic = name;
} else
{
if (_areSFXMuted)
return;
si.play(startTime, loops, volume * _sfxVolume, resumeTween);
}
```

```
} else
{
createNewSoundItem(name, volume, startTime, loops, resumeTween);
addAudio(name);
}
}

/**
* Pauses the specified sound.
*
* @param $name The string identifier of the sound
* @param $pauseTween A boolean that either pauses the fadeTween or allows it to continue
(default: true)
*
* @return void
*/
public function pauseSound( name:String, pauseTween:Boolean = true ):void
{
if (_soundsRef[name])
{
var si:SoundItem = SoundItem(_soundsRef[name]);
si.pause(pauseTween);
}
}

/**
* Stops the specified sound.
*
* @param $name The string identifier of the sound
*
* @return void
*/
public function stopSound( name:String, destroy:Boolean = true ):void
{
if (_soundsRef.hasOwnProperty(name))
{
var si:SoundItem = SoundItem(_soundsRef[name]);
si.stop();
//remove the music from memory
if (si.type == AudioEnum.TYPE_MUSIC)
{
if (destroy)
removeAudio(name);
}
//remove the voiceover from memory
if (si.type == AudioEnum.TYPE_VOICEOVER)
{
unhighlightSound(name, HIGHLIGHT_SCALAR); //reverse the volume change to highlight the
voiceover
if
```

```
(destroy)
removeAudio(name);
_currentVoiceOver = null;
}
}
}

/**
* Plays all the sounds that are in the sound dictionary.
*
* @param $resumeTweens A boolean that resumes all unfinished fade tweens (default: true)
* @param $useCurrentlyPlayingOnly A boolean that only plays the sounds which were currently playing before a pauseAllSounds() or stopAllSounds() call (default: false)
*
* @return void
*/
public function playAllSounds( resumeTweens:Boolean = true,
useCurrentlyPlayingOnly:Boolean = false ):void
{
var len:int = _sounds.length;
var si:SoundItem;
for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (useCurrentlyPlayingOnly)
{
if (si.pausedByAll)
{
si.pausedByAll = false;
playSound(si.name, si.volume, 0, 0, resumeTweens);
}
} else
{
playSound(si.name, si.volume, 0, 0, resumeTweens);
}
}
}

/**
* Pauses all the sounds that are in the sound dictionary.
*
* @param $pauseTweens A boolean that either pauses each SoundItem's fadeTween or allows them to continue (default: true)
* @param $useCurrentlyPlayingOnly A boolean that only pauses the sounds which are currently playing (default: true)
*
* @return void
*/
public function pauseAllSounds( pauseTweens:Boolean = true,
useCurrentlyPlayingOnly:Boolean = true ):void
{
```

```
var len:int = _sounds.length;
var si:SoundItem;
for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (useCurrentlyPlayingOnly)
{
if (!si.paused)
{
si.pausedByAll = true;
pauseSound(si.name, pauseTweens);
}
} else
{
pauseSound(si.name, pauseTweens);
}
}
}

/**
* Stops all the sounds that are in the sound dictionary.
*
* @param $useCurrentlyPlayingOnly A boolean that only stops the sounds which are currently playing (default: true)
*
* @return void
*/
public function stopAllSounds( useCurrentlyPlayingOnly:Boolean = true ):void
{
var len:int = _sounds.length;
var si:SoundItem;
for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (useCurrentlyPlayingOnly)
{
if (!si.paused)
{
si.pausedByAll = true;
stopSound(si.name);
}
} else
{
stopSound(si.name);
}
}
}

/**
*
```

```
Fades the sound to the specified volume over the specified amount of time.
*
* @param $name The string identifier of the sound
* @param $targVolume The target volume to fade to, between 0 and 1 (default: 0)
* @param $fadeLength The time to fade over, in seconds (default: 1)
* @param $stopOnComplete Added by Danny Miller from K2xL, stops the sound once the fade
is done if set to true
*
* @return void
*/
public function fadeSound( $name:String, $targVolume:Number = 0, $fadeLength:Number = 1,
$stopOnComplete:Boolean = false ):void
{
var si:SoundItem = SoundItem(_soundsRef[$name]);
si.fade($targVolume, $fadeLength, $stopOnComplete);
}

public function toggleSFXMute():void
{
_areSFXMuted = !_areSFXMuted;

var len:int = _sounds.length;
var si:SoundItem;

for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (si.type != AudioEnum.TYPE_MUSIC)
{
si.muted = _areSFXMuted;
if (_areSFXMuted)
si.setVolume(0);
else
si.setVolume(si.savedVolume);
}
}
}

public function toggleMusicMute():void
{
_isMusicMuted = !_isMusicMuted;

var len:int = _sounds.length;
var si:SoundItem;

for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (si.type == AudioEnum.TYPE_MUSIC)
{
si.muted
```

```
= _isMusicMuted;
if (_isMusicMuted)
si.setVolume(0);
else
si.setVolume(si.savedVolume);
if (_currentMusic == si.name)
{
if (_isMusicMuted)
stopSound(si.name, false);
else
playSound(si.name, si.volume, si.startTime, si.loops);
}
}
}
}

public function setSFXVolume( v:Number ):void
{
_sfxVolume = v;
var len:int = _sounds.length;
var si:SoundItem;

for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (si.type != AudioEnum.TYPE_MUSIC)
{
si.setVolume(si.savedVolume * _sfxVolume);
}
}
}

public function setMusicVolume( v:Number ):void
{
_musicVolume = v;

var len:int = _sounds.length;
var si:SoundItem;

for (var i:int = 0; i < len; i++)
{
si = _sounds[i];
if (si.type == AudioEnum.TYPE_MUSIC)
{
si.setVolume(si.savedVolume * _musicVolume);

if (_currentMusic == si.name)
playSound(si.name, si.volume, si.startTime, si.loops);
}
}
}
```

```
/**
* Sets the volume of the specified sound.
*
* @param $name The string identifier of the sound
* @param $volume The volume, between 0 and 1, to set the sound to
*
* @return void
*/
public function setSoundVolume( name:String, volume:Number ):void
{
var s:SoundItem;
if (_soundsRef[name])
_soundsRef[name].setVolume(volume);
}

/**
* Gets the position of the specified sound.
*
* @param $name The string identifier of the sound
*
* @return Number The current position of the sound, in milliseconds
*/
public function getSoundPosition( name:String ):Number
{
if (_soundsRef[name])
return _soundsRef[name].channel.position;
return 0;
}

/**
* Gets the SoundItem instance of the specified sound.
*
* @param $name The string identifier of the SoundItem
*
* @return SoundItem The SoundItem
*/
public function getSoundItem( name:String ):SoundItem
{
return SoundItem(_soundsRef[name]);
}

/**
* Reduces the volume of all other sounds to make the specified sound easier to hear
* @param name The name of the sound to highlight
* @param scalar The volume scalar to apply to sounds
*/
public function highlightSound( name:String, scalar:Number ):void
{
for
```

```
(var i:int = 0; i < _sounds.length; i++)
{
if (_sounds[i].name != name)
_sounds[i].setVolume(_sounds[i].volume * scalar);
}
_volumeScalar = scalar;
}

/**
* Reverts the highlighting that was done by a call to highlightSound
* @param name The name of the sound that was highlighted
* @param scalar The volume scalar to apply to sounds
*/
public function unhighlightSound( name:String, scalar:Number ):void
{
for (var i:int = 0; i < _sounds.length; i++)
{
if (_sounds[i].name != name)
_sounds[i].setVolume(_sounds[i].volume / scalar);
}
_volumeScalar = 1;
}

//- EVENT HANDLERS ------------------------------------------------------------------------------

/**
* Dispatched once a sound's fadeTween is completed if the sound was called to fade.
*/
private function handleFadeComplete( si:SoundItem ):void
{
//TODO dispatch a signal or do something when the fade is complete
}

/**
* Dispatched when a SoundItem has finished playback.
*/
private function handleSoundPlayComplete( si:SoundItem ):void
{
if (si.type == AudioEnum.TYPE_VOICEOVER)
{
//revert the volume highlighting and remove the audio from memory
unhighlightSound(si.name, HIGHLIGHT_SCALAR);
removeAudio(si.name);
_currentVoiceOver = null;
}
}

private function createNewSoundItem( name:String, volume:Number = 1, startTime:Number = 0,
loops:int = 0, resumeTween:Boolean = true ):SoundItem
{
var
```

```
si:SoundItem = ObjectPool.get(SoundItem);
si.init();
si.name = name;
si.position = 0;
si.paused = true;

//set the type of the audio and the volume
if (name.indexOf('sfx') > -1)
{
si.volume = (_areSFXMuted) ? 0 : volume;
si.muted = _areSFXMuted;
si.type = AudioEnum.TYPE_SFX;
} else if (name.indexOf('music') > 1)
{
si.volume = (_isMusicMuted) ? 0 : volume;
si.muted = _isMusicMuted;
si.type = AudioEnum.TYPE_MUSIC;
} else
{
si.volume = (_areSFXMuted) ? 0 : volume;
si.muted = _areSFXMuted;
si.type = AudioEnum.TYPE_VOICEOVER;
}

si.savedVolume = volume;
si.startTime = startTime;
si.loops = loops;
si.pausedByAll = false;

_soundsRef[name] = si;
_sounds.push(si);

return si;
}

//- GETTERS & SETTERS ----------------------------------------------------------------------------

/**
*
*/
public function get areSFXMuted():Boolean { return _areSFXMuted; }

public function get isMusicMuted():Boolean { return _isMusicMuted; }

public function get sfxVolume():Number { return _sfxVolume; }

public function get musicVolume():Number { return _musicVolume; }

//- HELPERS --------------------------------------------------------------------------------------

override
```

```
public function toString():String
{
return getQualifiedClassName(this);
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
}
}

-------------------------------------
File 42: igw\com\controller\sound\SoundItem.as
package com.controller.sound
{
import flash.events.Event;
import flash.events.EventDispatcher;
import flash.media.Sound;
import flash.media.SoundChannel;
import flash.media.SoundTransform;
import flash.utils.getQualifiedClassName;

import org.greensock.TweenLite;
import org.osflash.signals.Signal;

public class SoundItem extends EventDispatcher
{
public static const FADE_COMPLETE_SIGNAL:Signal = new Signal(SoundItem);
public static const PLAY_COMPLETE_SIGNAL:Signal = new Signal(SoundItem);

//- PRIVATE & PROTECTED VARIABLES
------------------------------------------------------------------------

private var _fadeTween:TweenLite;
private var _volume:Number;

//- PUBLIC & INTERNAL VARIABLES ----------------------------------------------------------------------------

public var channel:SoundChannel;
public var isLoaded:Boolean;
public var loading:Boolean;
public var loops:int;
public var muted:Boolean;
public var name:String;
public var paused:Boolean;
public var pausedByAll:Boolean;
public var position:int;
public var savedVolume:Number;
public var sound:Sound;
public var startTime:Number;
public var type:int;

//-
```

```
CONSTRUCTOR ----------------------------------------------------------------------------------------

public function SoundItem():void
{
super();
}

/**
*
*/
public function init():void
{
channel = new SoundChannel();
isLoaded = loading = false;
}

//- PRIVATE & PROTECTED METHODS
---------------------------------------------------------------------

/**
*
*/
private function fadeComplete( $stopOnComplete:Boolean ):void
{
if ($stopOnComplete)
stop();

FADE_COMPLETE_SIGNAL.dispatch(this);
}

//- PUBLIC & INTERNAL METHODS ----------------------------------------------------------------------

/**
* Plays the sound item.
*
* @param $startTime The time, in seconds, to start the sound at (default: 0)
* @param $loops The number of times to loop the sound (default: 0)
* @param $volume The volume to play the sound at (default: 1)
* @param $resumeTween If the sound volume is faded and while fading happens the sound is
stopped, this will resume that fade tween (default: true)
*
* @return void
*/
public function play( $startTime:Number = 0, $loops:int = 0, $volume:Number = 1,
$resumeTween:Boolean = true ):void
{
if (!paused)
return;

if (muted)
{
```

```
volume = 0;
} else
{
volume = $volume;
}
savedVolume = $volume;
startTime = $startTime;
loops = $loops;
paused = ($startTime == 0) ? true : false;

if (!paused)
position = startTime;
if (sound)
{
channel = sound.play(position, loops, new SoundTransform(volume));
if (channel)
{
channel.removeEventListener(Event.SOUND_COMPLETE, handleSoundComplete);
channel.addEventListener(Event.SOUND_COMPLETE, handleSoundComplete);
paused = false;

if ($resumeTween && (fadeTween != null))
fadeTween.resume();
}
}
}

/**
* Pauses the sound item.
*
* @param $pauseTween If a fade tween is happening at the moment the sound is paused, the
tween will be paused as well (default: true)
*
* @return void
*/
public function pause( $pauseTween:Boolean = true ):void
{
paused = true;
position = channel.position;
channel.stop();
channel.removeEventListener(Event.SOUND_COMPLETE, handleSoundComplete);

if ($pauseTween && (fadeTween != null))
fadeTween.pause();
}

/**
* Stops the sound item.
*
*
```

```
@return void
*/
public function stop():void
{
if (channel)
{
paused = true;
channel.stop();
channel.removeEventListener(Event.SOUND_COMPLETE, handleSoundComplete);
position = channel.position;
fadeTween = null;
}
}

/**
* Fades the sound item.
*
* @param $volume The volume to fade to (default: 0)
* @param $fadeLength The time, in seconds, to fade the sound (default: 1)
* @param $stopOnComplete Stops the sound once the fade is completed (default: false)
*
* @return void
*/
public function fade( $volume:Number = 0, $fadeLength:Number = 1,
$stopOnComplete:Boolean = false ):void
{
fadeTween = TweenLite.to(channel, $fadeLength, {volume:$volume, onComplete:fadeComplete,
onCompleteParams:[$stopOnComplete]});
}

/**
* Sets the volume of the sound item.
*
* @param $volume The volume, from 0 to 1, to set
*
* @return void
*/
public function setVolume( $volume:Number ):void
{
if (channel)
{
var curTransform:SoundTransform = channel.soundTransform;
if (curTransform)
{
curTransform.volume = $volume;
channel.soundTransform = curTransform;
}
_volume = $volume;
}
}

/**
```

```
* Clears the sound item for garbage collection.
*
* @return void
*/
public function destroy():void
{
if (channel)
channel.removeEventListener(Event.SOUND_COMPLETE, handleSoundComplete);
sound = null;
channel = null;
fadeTween = null;
}

//- EVENT HANDLERS ------------------------------------------------------------------------------

/**
*
*/
private function handleSoundComplete( $evt:Event ):void
{
stop();
PLAY_COMPLETE_SIGNAL.dispatch(this);
}

//- GETTERS & SETTERS ---------------------------------------------------------------------------

/**
*
*/
public function get volume():Number
{
if (channel && channel.soundTransform)
return channel.soundTransform.volume;

return 0;
}

/**
*
*/
public function set volume( $val:Number ):void
{
setVolume($val);
}

/**
*
*/
public function get fadeTween():TweenLite
{
```

```
return _fadeTween;
}

/**
*
*/
public function set fadeTween( $val:TweenLite ):void
{
if ($val == null)
TweenLite.killTweensOf(this);

_fadeTween = $val;
}

//- HELPERS ---------------------------------------------------------------------------------------

override public function toString():String
{
return getQualifiedClassName(this);
}

//- END CLASS -------------------------------------------------------------------------------------
}
}

-------------------------------------
File 43: igw\com\controller\toast\Toast.as
package com.controller.toast
{
import com.util.priorityqueue.IPrioritizable;

import org.parade.core.IView;

public class Toast implements IPrioritizable
{
private var _duration:Number;
private var _limit:int;
private var _priority:int;
private var _sound:String;
private var _state:String;
private var _type:Object;
private var _view:IView;

public function init( type:Object, view:IView ):void
{
_duration = type.duration;
_limit = type.limit;
_priority = type.priority;
_sound = type.sound;
_state
```

```
= type.state;
_type = type;
_view = view;
}

public function get duration():Number { return _duration; }
public function get limit():int { return _limit; }
public function get priority():int { return _priority; }
public function get sound():String { return _sound; }
public function get state():String { return _state; }
public function get type():Object { return _type; }
public function get view():IView { return _view; }

public function destroy():void
{
_type = null;
_view = null;
}
}
}

-------------------------------------
File 44: igw\com\controller\toast\ToastController.as
package com.controller.toast
{
import com.controller.fte.FTEController;
import com.controller.sound.SoundController;
import com.enum.ToastEnum;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.ToastEvent;
import com.event.TransactionEvent;
import com.model.prototype.IPrototype;
import com.model.transaction.TransactionVO;
import com.util.priorityqueue.IPriorityQueue;

import flash.events.Event;
import flash.events.IEventDispatcher;
import flash.events.TimerEvent;
import flash.utils.Dictionary;
import flash.utils.Timer;

import org.parade.core.IView;
import org.parade.core.ViewEvent;
import org.shared.ObjectPool;

public class ToastController
{
private var _currentToast:Toast;
private var _dispatcher:IEventDispatcher;
private
```

```
var _fteController:FTEController;
private var _limitLookup:Dictionary;
private var _soundController:SoundController;
private var _timer:Timer;
private var _toasts:IPriorityQueue;

[PostConstruct]
public function init():void
{
_dispatcher.addEventListener(StateEvent.GAME_BATTLE, onStateChange)
_dispatcher.addEventListener(StateEvent.GAME_SECTOR, onStateChange)
_dispatcher.addEventListener(StarbaseEvent.WELCOME_BACK, onStateChange)
_limitLookup = new Dictionary();
_timer = new Timer(0, 1);
_timer.addEventListener(TimerEvent.TIMER_COMPLETE, onTimerComplete);
_toasts = new ToastPriorityQueue();
}

public function addTransactionToast( prototype:IPrototype, transaction:TransactionVO ):void
{
if (prototype && transaction && !_fteController.running)
{
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.prototype = prototype;
toastEvent.toastType = transaction.type == TransactionEvent.STARBASE_REPAIR_FLEET ?
ToastEnum.FLEET_REPAIRED : ToastEnum.TRANSACTION_COMPLETE;
toastEvent.transaction = transaction;
_dispatcher.dispatchEvent(toastEvent);
}
}

public function addToast( type:Object, view:IView ):void
{
//if there is a limit to how many of these toasts we can have at once, keep track of it
if (type.limit > 0)
{
if (_limitLookup[type] == null)
_limitLookup[type] = 0;
if (_limitLookup[type] == type.limit)
return;
_limitLookup[type] += _limitLookup[type] + 1;
}
//create a new toast object for storage and tracking
var toast:Toast = ObjectPool.get(Toast);
toast.init(type, view);
//add the toast to the queue
_toasts.add(toast);
if (_currentToast == null)
showToast();
}

public
```

```
function killCurrentToast():void
{
if (_currentToast)
onTimerComplete(null);
}

private function onTimerComplete( e:TimerEvent ):void
{
_timer.reset()
//tell the toast to remove itself from the view stack
_currentToast.view.destroy();
//if there was a limit to this toast, reduce the count
if (_currentToast.limit > 0)
{
_limitLookup[_currentToast.type] = _limitLookup[_currentToast.type] - 1;
}
ObjectPool.give(_currentToast);
_currentToast = null;
//show the next toast
showToast();
}

private function showToast():void
{
_currentToast = Toast(_toasts.getNext());
if (_currentToast)
{
_timer.delay = _currentToast.duration;
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetView = _currentToast.view;
_dispatcher.dispatchEvent(viewEvent);
if (_currentToast.sound != null)
_soundController.playSound(_currentToast.sound, .49);
_timer.start();
}
}

private function onStateChange( e:Event ):void
{
if (!_fteController.running && !_timer.running && _currentToast == null)
showToast();
}

[Inject]
public function set dispatcher( v:IEventDispatcher ):void { _dispatcher = v; }
[Inject]
public function set fteController( v:FTEController ):void { _fteController = v; }
[Inject]
public function set soundController( v:SoundController ):void { _soundController = v; }
}
}
```

File 45: igw\com\controller\toast\ToastPriorityQueue.as
```
package com.controller.toast
{
import com.Application;
import com.event.StateEvent;
import com.util.priorityqueue.ArrayPriorityQueue;
import com.util.priorityqueue.IPrioritizable;

public class ToastPriorityQueue extends ArrayPriorityQueue
{
//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function ToastPriorityQueue()
{
super();
}

override public function getNext():IPrioritizable
{
if (isEmpty)
{
return null;
}

var toast:Toast;
for (var i:int = 0; i < _queue.length; i++)
{
toast = Toast(_queue[i]);
if ((Application.STATE != StateEvent.GAME_BATTLE || toast.state ==
StateEvent.GAME_BATTLE) && (toast.state == null || toast.state == Application.STATE))
{
_queue.splice(i, 1);
_map[toast] = null;
delete _map[toast];
return toast;
}
}

return null;
}
}
}
```

File 46: igw\com\controller\transaction\StarbaseBuildingBuildCommand.as
```
package
```

```
com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.StarbaseCategoryEnum;
import com.enum.TypeEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.ExternalInterfaceAPI;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuildingBuildCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var game:Game;
[Inject]
public var starbaseFactory:IStarbaseFactory;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var entity:Entity;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;
var vo:BuildingVO = starbaseModel.getBuildingByID(responseData.id);

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
entity = game.getEntity(clientData.id);
starbaseModel.updateBuildingID(clientData.id,
```

```
responseData.id);
if (entity)
{
vo = starbaseModel.getBuildingByID(responseData.id);
//update the id of the entity with the one from the server
game.updateEntityID(entity, responseData.id);
if (vo.itemClass == TypeEnum.PYLON)
{
var system:StarbaseSystem = StarbaseSystem(game.getSystem(StarbaseSystem));
if (system)
system.findPylonConnections(entity);
}
}
clientData.id = responseData.id;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (vo && vo.prototype)
{
if (vo.prototype.getValue("category") != StarbaseCategoryEnum.STARBASE_STRUCTURE)
{
toastController.addTransactionToast(vo.prototype, responseData);
var asset:AssetVO = AssetModel.instance.getEntityData(vo.asset);
if (asset)
{
var img:String = asset.mediumImage;
ExternalInterfaceAPI.shareTransaction(event.type, vo);
}
}
}
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
var id:String = (clientData) ? clientData.id : responseData.id;
vo = starbaseModel.removeBuildingByID(id);
entity = game.getEntity(id);
if (entity)
{
if (vo.itemClass == TypeEnum.PYLON)
{
system = StarbaseSystem(game.getSystem(StarbaseSystem));
if (system)
system.findPylonConnections(entity, true);
}
starbaseFactory.destroyStarbaseItem(entity);
}
transactionModel.removeTransaction(token);
break;
}
```

```
}
}
}
```

File 47: igw\com\controller\transaction\StarbaseBuildingMoveCommand.as

```
package com.controller.transaction
{
import com.enum.TypeEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.shared.grid.GridSystem;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuildingMoveCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var transactionModel:TransactionModel;
[Inject]
public var game:Game;
[Inject]
public var starbaseFactory:IStarbaseFactory;
[Inject]
public var starbaseModel:StarbaseModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var entity:Entity;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;
var vo:BuildingVO;
switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
break;
```

```
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
var id:String = clientData.id;
vo = starbaseModel.getBuildingByID(id);
//remove the building vo from the grid
starbaseModel.grid.removeFromGrid(vo);
//update building vo position
vo.baseX = clientData.baseX;
vo.baseY = clientData.baseY;
//add back to the grid
starbaseModel.grid.addToGrid(vo, true);
entity = game.getEntity(id);
if (entity)
{
var position:Position = entity.get(Position);
//update the entity position
starbaseModel.grid.convertBuildingGridToIso(position.position, vo);
//have to do this to update followers
position.x = position.position.x;
position.y = position.position.y;

var system:StarbaseSystem = StarbaseSystem(game.getSystem(StarbaseSystem));
if (system)
{
if (vo.itemClass == TypeEnum.PYLON)
{
system.positionPylonBase(entity);
system.findPylonConnections(entity);
}
system.depthSort(StarbaseSystem.DEPTH_SORT_ALL);
}

//update the entity on the grid
var gridSystem:GridSystem = GridSystem(game.getSystem(GridSystem));
if (gridSystem)
gridSystem.forceGridCheck(entity);
}
transactionModel.removeTransaction(token);
break;
}
}
}
}
```

File 48: igw\com\controller\transaction\StarbaseBuildingRecycleCommand.as

```
package com.controller.transaction
{
import com.enum.StarbaseConstructionEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuildingRecycleCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var game:Game;
[Inject]
public var starbaseFactory:IStarbaseFactory;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var buildingVO:BuildingVO = clientData.buildingVO;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;

case StarbaseTransactionStateEnum.TIMER_DONE:
transactionModel.removeTransaction(responseData.token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
if (clientData)
{
//add the building back to the starbase
starbaseModel.addBuilding(buildingVO);
```

```
var starbaseSystem:StarbaseSystem = StarbaseSystem(game.getSystem(StarbaseSystem));
if (starbaseSystem)
{
if (buildingVO.constructionCategory == StarbaseConstructionEnum.PLATFORM)
{
starbaseFactory.createBaseItem(buildingVO.id, buildingVO);
if (starbaseSystem)
starbaseSystem.depthSort(StarbaseSystem.DEPTH_SORT_PLATFORMS);
} else
{
starbaseFactory.createBuilding(buildingVO.id, buildingVO);
if (starbaseSystem)
starbaseSystem.depthSort(StarbaseSystem.DEPTH_SORT_BUILDINGS);
}

if (starbaseSystem)
starbaseSystem.findPylonConnections();
}
}
transactionModel.removeTransaction(responseData.token);
break;
}
}
}
}

------------------------------------
File 49: igw\com\controller\transaction\StarbaseBuildingUpgradeCommand.as
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.TypeEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.ExternalInterfaceAPI;

import org.ash.core.Entity;
import org.ash.core.Game;
import
```

```
org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuildingUpgradeCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var game:Game;
[Inject]
public var prototypeModel:PrototypeModel;
[Inject]
public var starbaseFactory:IStarbaseFactory;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var entity:Entity;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;
var vo:BuildingVO = starbaseModel.getBuildingByID(responseData.id);
var upgradeVO:IPrototype;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
{
if (vo)
{
toastController.addTransactionToast(vo.prototype, responseData);
var sis:StarbaseInteractSystem =
StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem));
if (sis)
{
entity = game.getEntity(responseData.id);
if (entity)
{
starbaseFactory.updateStarbaseBuilding(entity);
```

```
if (vo.itemClass == TypeEnum.PYLON)
{
var starbaseSystem:StarbaseSystem = StarbaseSystem(game.getSystem(StarbaseSystem));
if (starbaseSystem)
starbaseSystem.findPylonConnections(game.getEntity(vo.id));
}
}
sis.updateRanges();
}
var asset:AssetVO = AssetModel.instance.getEntityData(vo.asset);
if (asset)
{
if (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER)
ExternalInterfaceAPI.shareTransaction(event.type, vo);
}
}
transactionModel.removeTransaction(token);
break;
}
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
entity = game.getEntity(responseData.id);
if (entity)
{
starbaseFactory.updateStarbaseBuilding(entity);
}
transactionModel.removeTransaction(token);
break;
}

}
}
}

-----------------------------------
File 50: igw\com\controller\transaction\StarbaseBuildShipCommand.as
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.fleet.FleetModel;
import com.model.fleet.ShipVO;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.ExternalInterfaceAPI;

import org.robotlegs.extensions.presenter.impl.Command;

public
```

```
class StarbaseBuildShipCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
if(event == null)
return;

if(fleetModel == null)
return;

if(transactionModel == null)
return;

var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;

if(responseData == null)
return;

var ship:ShipVO = fleetModel.getShip(responseData.id);
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
if(clientData != null)
fleetModel.updateShipID(clientData.id, responseData.id);
break;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
if (ship)
ship.built = false;
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (ship)
{
toastController.addTransactionToast(ship.prototypeVO, responseData);
ExternalInterfaceAPI.shareTransaction(responseData.type, ship);
ship.built = true;
}
transactionModel.removeTransaction(token);
break;
```

```
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
fleetModel.removeShip(responseData.id);
transactionModel.removeTransaction(token);
break;
}

}
}
}
```

File 51: igw\com\controller\transaction\StarbaseBuyOtherStoreItemCommand.as

```
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.prototype.IPrototype;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuyOtherStoreItemCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
break;
```

```
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (clientData)
toastController.addTransactionToast(clientData.prototype, responseData);
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(token);
break;
}
}
}
}

-----------------------------------
File 52: igw\com\controller\transaction\StarbaseBuyResourcesCommand.as
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuyResourcesCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;

switch (responseData.state)
{
```

```
case StarbaseTransactionStateEnum.SAVED:
break;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (clientData)
toastController.addTransactionToast(clientData.prototype, responseData);
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(token);
break;
}
}
}
}

-----------------------------------
File 53: igw\com\controller\transaction\StarbaseBuyStoreItemCommand.as
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.prototype.IPrototype;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseBuyStoreItemCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var
```

```
prototype:IPrototype = (clientData != null) ? clientData.buff :
starbaseModel.getBuffByID(responseData.id);
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
if(clientData == null)
{
// Possible if the client transaction was created by server and not registered in client.
return;
}
if (clientData.id)
starbaseModel.updateBuffID(clientData.id, responseData.id);
if (prototype == null)
prototype = starbaseModel.getBuffByID(responseData.id);
//if (responseData.timeMS > 0)
toastController.addTransactionToast(prototype, responseData);
transactionController.dataImported();
break;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (responseData.timeMS > 0)
toastController.addTransactionToast(prototype, responseData);
starbaseModel.removeBuffByID(responseData.id);
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
if(clientData == null)
{
// Possible if the client transaction was created by server and not registered in client.
return;
}
//try to remove the client buff if it hasn't been updated yet
if (clientData.id)
starbaseModel.removeBuffByID(clientData.id);
starbaseModel.removeBuffByID(responseData.id);
transactionModel.removeTransaction(token);
break;
}
}
}
}

------------------------------------
File 54: igw\com\controller\transaction\StarbaseCancelContractCommand.as
package com.controller.transaction
{
import
```

```
com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseCancelContractCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(token);
break;
}

}
}
}
```

File 55: igw\com\controller\transaction\StarbaseContractCommand.as

```
package com.controller.transaction
{
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import
```

```
com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseContractCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var transactionModel:TransactionModel;
[Inject]
public var starbaseModel:StarbaseModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
break;
}
}
}
}
```

File 56: igw\com\controller\transaction\StarbaseNegotiateContractCommand.as

```
package com.controller.transaction
{
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseNegotiateContractCommand extends Command
{
[Inject]
public
```

```
var event:TransactionEvent;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
if (clientData && clientData.callback)
clientData.callback();
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
if (clientData && clientData.callback)
clientData.callback();
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
if (clientData && clientData.callback)
clientData.callback();
transactionModel.removeTransaction(token);
break;
}

}
}
}



-----------------------------------
File 57: igw\com\controller\transaction\StarbaseRecallFleetCommand.as
package com.controller.transaction
{
import com.enum.FleetStateEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.enum.ui.ButtonEnum;
import com.event.TransactionEvent;
import com.model.fleet.FleetModel;
import
```

```
com.model.fleet.FleetVO;
import com.model.transaction.TransactionVO;
import com.ui.alert.ConfirmationView;
import com.ui.core.ButtonPrototype;
import com.ui.core.ViewFactory;

import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseRecallFleetCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var viewFactory:IViewFactory;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var fleet:FleetVO = fleetModel.getFleet(responseData.id);
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
break;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
break;
case StarbaseTransactionStateEnum.CANCELLED:
break;
case StarbaseTransactionStateEnum.FAILED:
if (fleet && fleet.inBattle)
{
fleet.state = FleetStateEnum.OUT;
fleetModel.updateFleet(fleet);
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype('OK', null, null, true, ButtonEnum.RED_A));

var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
var view:ConfirmationView = ConfirmationView(viewFactory.createView(ConfirmationView));

view.setup('IN COMBAT', "The currently selected fleet is in combat and cannot recall.", buttons)
viewEvent.targetView = view;
dispatch(viewEvent);
}
```

```
break;
}
}
}
}

-----------------------------------
File 58: igw\com\controller\transaction\StarbaseRecycleShipCommand.as
package com.controller.transaction
{
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.fleet.FleetModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseRecycleShipCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
break;
case StarbaseTransactionStateEnum.TIMER_RUNNING:
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (clientData)
fleetModel.removeShip(clientData.id);
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(responseData.token);
break;
}
}
}
}

-----------------------------------
```

File 59: igw\com\controller\transaction\StarbaseRefitBuildingCommand.as

```
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.PurchaseTypeEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.ExternalInterfaceAPI;

import flash.utils.Dictionary;

import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseRefitBuildingCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var game:Game;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var starbaseFactory:IStarbaseFactory;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;
var buildingVO:BuildingVO = starbaseModel.getBuildingByID(responseData.id);
var system:StarbaseInteractSystem =
StarbaseInteractSystem(game.getSystem(StarbaseInteractSystem));

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
transactionModel.updatedTransaction(responseData);
break;
```

```
case StarbaseTransactionStateEnum.TIMER_DONE:
toastController.addTransactionToast(buildingVO.prototype, responseData);
ExternalInterfaceAPI.shareTransaction(responseData.type, buildingVO);
buildingVO.refitModules = new Dictionary();
starbaseFactory.updateStarbaseBuilding(game.getEntity(buildingVO.id));
transactionModel.removeTransaction(token);
//hide any residual ranges
if (system)
system.updateRanges();
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
if (clientData && clientData.purchaseType == PurchaseTypeEnum.INSTANT)
{
//rollback the modules on this building
buildingVO.refitModules = new Dictionary();
starbaseFactory.updateStarbaseBuilding(game.getEntity(buildingVO.id));
}
transactionModel.removeTransaction(token);
//hide any residual ranges
if (system)
system.updateRanges();
break;
}
}
}
}

-----------------------------------
File 60: igw\com\controller\transaction\StarbaseRefitShipCommand.as
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.fleet.FleetModel;
import com.model.fleet.ShipVO;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.ExternalInterfaceAPI;

import flash.utils.Dictionary;

import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseRefitShipCommand extends Command
{
[Inject]
public
```

```
var event:TransactionEvent;
[Inject]
public var game:Game;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var ship:ShipVO = fleetModel.getShip(responseData.id);
var token:int = event.transactionToken;

if(ship == null)
return;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
fleetModel.removeShipFromFleet(responseData.id);
case StarbaseTransactionStateEnum.TIMER_RUNNING:
ship.built = false;
ship.refiting = true;
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
toastController.addTransactionToast(ship.prototypeVO, responseData);
ship.built = true;
ship.refiting = false;
ship.refitModules = new Dictionary();
ship.calculateCosts();
ExternalInterfaceAPI.shareTransaction(responseData.type, ship);
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
ship.built = true;
ship.refiting = false;
ship.calculateCosts();
transactionModel.removeTransaction(token);
break;
}
}
}
}
```

File 61: igw\com\controller\transaction\StarbaseRepairBaseCommand.as

```
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.TypeEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseRepairBaseCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var game:Game;
[Inject]
public var starbaseFactory:IStarbaseFactory;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var entity:Entity;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;
var vo:BuildingVO = starbaseModel.getBuildingByID(responseData.id, false);

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
if (vo)
{
if (vo.currentHealth == 0)
{
vo.currentHealth
```

```
= .1;
entity = game.getEntity(vo.id);
if (entity)
starbaseFactory.updateStarbaseBuilding(entity);
}
}
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if (vo && vo.itemClass == TypeEnum.PYLON)
{
vo.currentHealth = 1;
var system:StarbaseSystem = StarbaseSystem(game.getSystem(StarbaseSystem));
if (system)
system.findPylonConnections(game.getEntity(vo.id));
}
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(token);
break;
}
}
}
}

-----------------------------------
File 62: igw\com\controller\transaction\StarbaseRepairFleetCommand.as
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseRepairFleetCommand extends Command
{

[Inject]
public var event:TransactionEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var toastController:ToastController;
[Inject]
```

```
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var fleet:FleetVO = fleetModel.getFleet(responseData.id);
var token:int = event.transactionToken;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
fleetModel.repairFleet(fleet, (responseData.timeRemainingMS <= 0));
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
toastController.addTransactionToast(fleet, responseData);
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(token);
break;
}
}
}
}
```

File 63: igw\com\controller\transaction\StarbaseResearchCommand.as

```
package com.controller.transaction
{
import com.controller.toast.ToastController;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.player.CurrentUser;
import com.model.starbase.BaseVO;
import com.model.starbase.ResearchVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.ExternalInterfaceAPI;
import com.service.server.incoming.data.ResearchData;

import org.robotlegs.extensions.presenter.impl.Command;
import org.shared.ObjectPool;

public
```

```
class StarbaseResearchCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var toastController:ToastController;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
if(event == null)
return;

if(starbaseModel == null)
return;

if(transactionModel == null)
return;

var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;

if(responseData == null)
return;

switch (responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
if(clientData == null)
return;

var baseVO:BaseVO = clientData.baseVO;
var researchData:ResearchData = ObjectPool.get(ResearchData);

if(baseVO == null)
return;
if(researchData == null)
return;

researchData.baseID = baseVO.id;
researchData.id = responseData.id;
researchData.playerOwnerID = CurrentUser.id;
researchData.prototype = clientData.vo;
starbaseModel.importResearchData(researchData);
transactionModel.updatedTransaction(responseData);
ObjectPool.give(researchData);
break;
case
```

```
StarbaseTransactionStateEnum.TIMER_RUNNING:
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
if(toastController == null)
return;
var research:ResearchVO = starbaseModel.getResearchByID(responseData.id);
if(research == null)
return;
toastController.addTransactionToast(research, responseData);
ExternalInterfaceAPI.shareTransaction(event.type, research);
transactionModel.removeTransaction(responseData.token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(responseData.token);
starbaseModel.removeResearchByID(responseData.id);
break;
}
}
}
}

------------------------------------
File 64: igw\com\controller\transaction\StarbaseUpdateFleetCommand.as
package com.controller.transaction
{
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.fleet.FleetModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import org.robotlegs.extensions.presenter.impl.Command;

public class StarbaseUpdateFleetCommand extends Command
{
[Inject]
public var event:TransactionEvent;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var transactionModel:TransactionModel;

override public function execute():void
{
var clientData:Object = event.clientData;
var responseData:TransactionVO = event.responseData;
var token:int = event.transactionToken;

switch
```

```
(responseData.state)
{
case StarbaseTransactionStateEnum.SAVED:
case StarbaseTransactionStateEnum.TIMER_RUNNING:
transactionModel.updatedTransaction(responseData);
break;
case StarbaseTransactionStateEnum.TIMER_DONE:
transactionModel.removeTransaction(token);
break;
case StarbaseTransactionStateEnum.CANCELLED:
case StarbaseTransactionStateEnum.FAILED:
transactionModel.removeTransaction(token);
break;
}
}
}
}

-------------------------------------
File 65: igw\com\controller\transaction\TransactionController.as
ï»¿package com.controller.transaction
{
import com.controller.ServerController;
import com.controller.transaction.requirements.BlueprintRequirement;
import com.controller.transaction.requirements.BuildingLevelRequirement;
import com.controller.transaction.requirements.BuildingNotBusyRequirement;
import com.controller.transaction.requirements.BuildingNotDamagedRequirement;
import com.controller.transaction.requirements.CategoryNotBuildingRequirement;
import com.controller.transaction.requirements.IRequirementFactory;
import com.controller.transaction.requirements.RequirementVO;
import com.controller.transaction.requirements.ResearchRequirement;
import com.controller.transaction.requirements.TechNotKnownRequirement;
import com.controller.transaction.requirements.UnderMaxCountRequirement;
import com.controller.transaction.requirements.UnderMaxResourceRequirement;
import com.controller.transaction.requirements.UniqueEquipRequirement;
import com.enum.CurrencyEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.PurchaseTypeEnum;
import com.enum.server.RequestEnum;
import com.enum.server.StarbaseBuildStateEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.event.TransactionEvent;
import com.model.asset.AssetModel;
import com.model.blueprint.BlueprintVO;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BaseVO;
import
```

```
com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.IShipyardPresenter;
import com.service.server.ITransactionRequest;
import com.service.server.ITransactionResponse;
import com.service.server.outgoing.starbase.StarbaseBuildNewBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseBuildShipRequest;
import com.service.server.outgoing.starbase.StarbaseBuyOtherStoreItemRequest;
import com.service.server.outgoing.starbase.StarbaseBuyResourceRequest;
import com.service.server.outgoing.starbase.StarbaseBuyStoreItemRequest;
import com.service.server.outgoing.starbase.StarbaseBuyoutBlueprintRequest;
import com.service.server.outgoing.starbase.StarbaseCancelContractRequest;
import com.service.server.outgoing.starbase.StarbaseCancelTransactionRequest;
import com.service.server.outgoing.starbase.StarbaseCompleteBlueprintResearchRequest;
import com.service.server.outgoing.starbase.StarbaseInstancedMissionStartRequest;
import com.service.server.outgoing.starbase.StarbaseLaunchFleetRequest;
import com.service.server.outgoing.starbase.StarbaseMintNFTRequest;
import com.service.server.outgoing.starbase.StarbaseMissionAcceptRequest;
import com.service.server.outgoing.starbase.StarbaseMissionAcceptRewardsRequest;
import com.service.server.outgoing.starbase.StarbaseMissionStepRequest;
import com.service.server.outgoing.starbase.StarbaseMoveBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseMoveStarbaseRequest;
import com.service.server.outgoing.starbase.StarbaseMoveStarbaseToTransgateRequest;
import com.service.server.outgoing.starbase.StarbaseNegotiateContractRequest;
import com.service.server.outgoing.starbase.StarbaseRecallFleetRequest;
import com.service.server.outgoing.starbase.StarbaseRecycleBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseRecycleShipRequest;
import com.service.server.outgoing.starbase.StarbaseRefitBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseRefitShipRequest;
import com.service.server.outgoing.starbase.StarbaseRenameFleetRequest;
import com.service.server.outgoing.starbase.StarbaseRenamePlayerRequest;
import com.service.server.outgoing.starbase.StarbaseRepairBaseRequest;
import com.service.server.outgoing.starbase.StarbaseRepairFleetRequest;
import com.service.server.outgoing.starbase.StarbaseRerollBlueprintChanceRequest;
import com.service.server.outgoing.starbase.StarbaseRerollBlueprintReceivedRequest;
import com.service.server.outgoing.starbase.StarbaseResearchRequest;
import com.service.server.outgoing.starbase.StarbaseSpeedUpTransactionRequest;
import com.service.server.outgoing.starbase.StarbaseUpdateFleetRequest;
import com.service.server.outgoing.starbase.StarbaseUpgradeBuildingRequest;
import com.util.statcalc.StatCalcUtil;

import flash.events.IEventDispatcher;
import flash.utils.Dictionary;
import flash.net.navigateToURL;
import flash.net.URLRequest;

import org.as3commons.logging.api.ILogger;
import
```

```
org.as3commons.logging.api.getLogger;
import org.robotlegs.extensions.eventCommandMap.api.IEventCommandMap;

public class TransactionController
{
private var _assetModel:AssetModel;
private var _commandMap:IEventCommandMap;
private var _eventDispatcher:IEventDispatcher;
private var _prototypeModel:PrototypeModel;
private var _requirementVO:RequirementVO;
private var _serverController:ServerController;
private var _starbaseModel:StarbaseModel;
private var _transactionModel:TransactionModel;
private var _reqFactory:IRequirementFactory;
private var _shipyardPresenter:IShipyardPresenter;

protected const _logger:ILogger = getLogger('TransactionController');

[PostConstruct]
public function init():void
{
_commandMap.map(TransactionEvent.STARBASE_BUILDING_BUILD,
TransactionEvent).toCommand(StarbaseBuildingBuildCommand);
_commandMap.map(TransactionEvent.STARBASE_BUILDING_MOVE,
TransactionEvent).toCommand(StarbaseBuildingMoveCommand);
_commandMap.map(TransactionEvent.STARBASE_BUILDING_RECYCLE,
TransactionEvent).toCommand(StarbaseBuildingRecycleCommand);
_commandMap.map(TransactionEvent.STARBASE_BUILD_SHIP,
TransactionEvent).toCommand(StarbaseBuildShipCommand);
_commandMap.map(TransactionEvent.STARBASE_BUILDING_UPGRADE,
TransactionEvent).toCommand(StarbaseBuildingUpgradeCommand);
_commandMap.map(TransactionEvent.STARBASE_REFIT_BUILDING,
TransactionEvent).toCommand(StarbaseRefitBuildingCommand);
_commandMap.map(TransactionEvent.STARBASE_REFIT_SHIP,
TransactionEvent).toCommand(StarbaseRefitShipCommand);
_commandMap.map(TransactionEvent.STARBASE_REPAIR_BASE,
TransactionEvent).toCommand(StarbaseRepairBaseCommand);
_commandMap.map(TransactionEvent.STARBASE_RESEARCH,
TransactionEvent).toCommand(StarbaseResearchCommand);
_commandMap.map(TransactionEvent.STARBASE_UPDATE_FLEET,
TransactionEvent).toCommand(StarbaseUpdateFleetCommand);
_commandMap.map(TransactionEvent.STARBASE_REPAIR_FLEET,
TransactionEvent).toCommand(StarbaseRepairFleetCommand);
_commandMap.map(TransactionEvent.STARBASE_RECALL_FLEET,
TransactionEvent).toCommand(StarbaseRecallFleetCommand);
_commandMap.map(TransactionEvent.STARBASE_RECYCLE_SHIP,
TransactionEvent).toCommand(StarbaseRecycleShipCommand);
_commandMap.map(TransactionEvent.STARBASE_NEGOTIATE_CONTRACT_REQUEST,
TransactionEvent).toCommand(StarbaseNegotiateContractCommand);
_commandMap.map(TransactionEvent.STARBASE_CANCEL_CONTRACT_REQUEST,
TransactionEvent).toCommand(StarbaseCancelContractCommand);
```

```
_commandMap.map(TransactionEvent.STARBASE_BUY_RESOURCES,
TransactionEvent).toCommand(StarbaseBuyResourcesCommand);
_commandMap.map(TransactionEvent.STARBASE_BUY_STORE_ITEM,
TransactionEvent).toCommand(StarbaseBuyStoreItemCommand);
_commandMap.map(TransactionEvent.STARBASE_BUY_OTHER_STORE_ITEM,
TransactionEvent).toCommand(StarbaseBuyOtherStoreItemCommand);

_requirementVO = new RequirementVO(_assetModel);
}

public function addTransaction( request:ITransactionRequest, id:String, type:String, data:Object,
createDefaultTransaction:Boolean = true ):void
{
if (!data)
data = {};
_transactionModel.addTransaction(request, id, type, data, createDefaultTransaction);
_logger.debug('Adding New Transaction Type: {0}, Token: {1}', [type, request.token]);
_serverController.send(request);
}

public function handleResponse( response:ITransactionResponse ):void
{
_logger.debug('Response Transaction Token: {}', response.token);
var clientData:Object = _transactionModel.handleResponse(response);
dispatch(response.token, clientData, response.data);
}

private function getRequest( protocolID:int, header:int ):ITransactionRequest
{
return ITransactionRequest(_serverController.getRequest(protocolID, header));
}

private function dispatch( transactionToken:int, clientData:Object, responseData:TransactionVO
):void
{
if (responseData.type)
_eventDispatcher.dispatchEvent(new TransactionEvent(responseData.type, transactionToken,
clientData, responseData));
else
{
//if we don't know what the transaction is (due to a missing type), remove it
_transactionModel.removeTransaction(responseData.token);
}
}


//==========================================================================
//**************************************************************************************************
// NEW TRANSACTION
//**************************************************************************************************
```

```
//=============================================================================

public function starbaseBuild( buildingVO:BuildingVO, purchaseType:uint ):Boolean
{
if (checkCost(buildingVO, purchaseType))
{
var buildNewBuildingRequest:StarbaseBuildNewBuildingRequest =
StarbaseBuildNewBuildingRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BUILD_NEW_BUILDING));
buildNewBuildingRequest.buildingPrototype = buildingVO.name;
buildNewBuildingRequest.locationX = buildingVO.baseX;
buildNewBuildingRequest.locationY = buildingVO.baseY;
buildNewBuildingRequest.centerSpaceBase = false;
buildNewBuildingRequest.purchaseType = purchaseType;

buildNewBuildingRequest.expectedCost.time_cost_milliseconds = (purchaseType ==
PurchaseTypeEnum.INSTANT) ? 0 : buildingVO.buildTimeSeconds * 1000;
buildNewBuildingRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
buildNewBuildingRequest.expectedCost.energyCost =
_requirementVO.purchaseVO.energyCost;
buildNewBuildingRequest.expectedCost.syntheticCost =
_requirementVO.purchaseVO.syntheticCost;
buildNewBuildingRequest.expectedCost.creditsCost =
_requirementVO.purchaseVO.creditsCost;

if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
var premiumCost:int;
if (purchaseType == PurchaseTypeEnum.INSTANT)
premiumCost = _requirementVO.purchaseVO.premium;
else
premiumCost = _requirementVO.purchaseVO.resourcePremiumCost;

buildNewBuildingRequest.expectedCost.hardCurrencyCost = premiumCost;
}

buildingVO.buildState = (purchaseType == PurchaseTypeEnum.INSTANT) ?
StarbaseBuildStateEnum.DONE : StarbaseBuildStateEnum.BUILDING;
addTransaction(buildNewBuildingRequest, buildingVO.id,
TransactionEvent.STARBASE_BUILDING_BUILD, {id:buildingVO.id,
purchaseType:purchaseType});
return true;
} else
_logger.warn("Build request on {} failed to send to the server due to insufficient funds",
buildingVO.itemClass);
return false;
}

public
```

```
function starbaseUpgradeBuilding( buildingVO:BuildingVO, upgradeVO:IPrototype,
purchaseType:uint ):Boolean
{
if (checkCost(upgradeVO, purchaseType))
{
var upgradeBuildingRequest:StarbaseUpgradeBuildingRequest =
StarbaseUpgradeBuildingRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_UPGRADE_BUILDING));
upgradeBuildingRequest.buildingPersistence = buildingVO.id;
upgradeBuildingRequest.purchaseType = purchaseType;

upgradeBuildingRequest.expectedCost.time_cost_milliseconds = (purchaseType ==
PurchaseTypeEnum.INSTANT) ? 0 : upgradeVO.buildTimeSeconds * 1000;
upgradeBuildingRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
upgradeBuildingRequest.expectedCost.energyCost =
_requirementVO.purchaseVO.energyCost;
upgradeBuildingRequest.expectedCost.syntheticCost =
_requirementVO.purchaseVO.syntheticCost;
upgradeBuildingRequest.expectedCost.creditsCost = _requirementVO.purchaseVO.creditsCost;

if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
var premiumCost:int;
if (purchaseType == PurchaseTypeEnum.INSTANT)
premiumCost = _requirementVO.purchaseVO.premium;
else
premiumCost = _requirementVO.purchaseVO.resourcePremiumCost;

upgradeBuildingRequest.expectedCost.hardCurrencyCost = premiumCost;
}

buildingVO.buildState = (purchaseType == PurchaseTypeEnum.INSTANT) ?
StarbaseBuildStateEnum.DONE : StarbaseBuildStateEnum.UPGRADING;
addTransaction(upgradeBuildingRequest, buildingVO.id,
TransactionEvent.STARBASE_BUILDING_UPGRADE, {id:buildingVO.id,
purchaseType:purchaseType});
return true;
} else
_logger.warn("Upgrade request on {} failed to send to the server due to insufficient funds",
buildingVO.itemClass);
return false;
}

public function starbaseMoveBuilding( buildingVO:BuildingVO, oldBaseX:int, oldBaseY:int ):void
{
var moveBuildingRequest:StarbaseMoveBuildingRequest =
StarbaseMoveBuildingRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MOVE_BUILDING));
moveBuildingRequest.buildingKey = buildingVO.id;
moveBuildingRequest.locationX
```

```
= buildingVO.baseX;
moveBuildingRequest.locationY = buildingVO.baseY;
moveBuildingRequest.centerSpaceBase = false;
addTransaction(moveBuildingRequest, buildingVO.id,
TransactionEvent.STARBASE_BUILDING_MOVE, {id:buildingVO.id, baseX:oldBaseX,
baseY:oldBaseY}, false);
}

public function starbaseRecycleBuilding( buildingVO:BuildingVO ):void
{
var recycleBuildingRequest:StarbaseRecycleBuildingRequest =
StarbaseRecycleBuildingRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_RECYCLE_BUILDING));
recycleBuildingRequest.buildingPersistence = buildingVO.id;
addTransaction(recycleBuildingRequest, buildingVO.id,
TransactionEvent.STARBASE_BUILDING_RECYCLE, {id:buildingVO.id,
buildingVO:buildingVO});
}

/**
* Send a request to the sever to refit the building with the new modules
*
* @param buildingVO The vo of the building to refit
* @param equipped The new modules that we want on the building
* @param currentModules Save a reference to the old modules so that we can roll back in case
the server denies the refit request
* @param instant Should the refit be done instantly or not
*/
public function starbaseRefitBuilding( buildingVO:BuildingVO, modules:Dictionary,
purchaseType:uint ):void
{
if (checkCost(buildingVO, purchaseType))
{
var refitBuildingRequest:StarbaseRefitBuildingRequest =
StarbaseRefitBuildingRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_REFIT_BUILDING));
refitBuildingRequest.buildingPersistence = buildingVO.id;
refitBuildingRequest.purchaseType = purchaseType;
refitBuildingRequest.modules = modules;
refitBuildingRequest.slots = buildingVO.prototype.getValue('slots');

refitBuildingRequest.expectedCost.time_cost_milliseconds = (purchaseType ==
PurchaseTypeEnum.INSTANT) ? 0 : buildingVO.refitBuildTimeSeconds * 1000;
refitBuildingRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
refitBuildingRequest.expectedCost.energyCost = _requirementVO.purchaseVO.energyCost;
refitBuildingRequest.expectedCost.syntheticCost = _requirementVO.purchaseVO.syntheticCost;
refitBuildingRequest.expectedCost.creditsCost = _requirementVO.purchaseVO.creditsCost;

if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
```

```
var premiumCost:int;
if (purchaseType == PurchaseTypeEnum.INSTANT)
premiumCost += _requirementVO.purchaseVO.premium;
else
premiumCost += _requirementVO.purchaseVO.resourcePremiumCost;

refitBuildingRequest.expectedCost.hardCurrencyCost = premiumCost;
}

addTransaction(refitBuildingRequest, buildingVO.id,
TransactionEvent.STARBASE_REFIT_BUILDING, {id:buildingVO.id,
modules:buildingVO.modules, purchaseType:purchaseType});
}
}

public function starbaseRepairBuildings( hardCurrency:int, purchaseType:uint ):void
{
var repairTransactionRequest:StarbaseRepairBaseRequest =
StarbaseRepairBaseRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_REPAIR_BASE));
repairTransactionRequest.purchaseType = purchaseType;
repairTransactionRequest.centerSpaceBase = false;

//expected costs are currently not used on building repair
/*repairTransactionRequest.expectedCost.time_cost_milliseconds = 0;
repairTransactionRequest.expectedCost.alloyCost = 0;
repairTransactionRequest.expectedCost.energyCost = 0;
repairTransactionRequest.expectedCost.syntheticCost = 0;
repairTransactionRequest.expectedCost.creditsCost = 0;*/

addTransaction(repairTransactionRequest, '', TransactionEvent.STARBASE_REPAIR_BASE, {},
false);
}

public function dockUpdateFleet( selectedFleet:FleetVO ):void
{
var updateFleetRequest:StarbaseUpdateFleetRequest =
StarbaseUpdateFleetRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_UPDATE_FLEET));
updateFleetRequest.fleet = selectedFleet;
addTransaction(updateFleetRequest, selectedFleet.id,
TransactionEvent.STARBASE_UPDATE_FLEET, {vo:selectedFleet}, false);
}

public function dockChangeFleetName( fleetID:String, newName:String ):void
{
var renameFleetRequest:StarbaseRenameFleetRequest =
StarbaseRenameFleetRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_RENAME_FLEET));
renameFleetRequest.fleetId
```

```
= fleetID;
renameFleetRequest.newName = newName;
addTransaction(renameFleetRequest, fleetID,
TransactionEvent.STARBASE_RENAME_FLEET, {id:fleetID}, false);
}

public function dockRepairShip( selectedFleet:FleetVO, purchaseType:uint ):void
{
if (checkCost(selectedFleet, purchaseType))
{
var repairFleetRequest:StarbaseRepairFleetRequest =
StarbaseRepairFleetRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_REPAIR_FLEET));
repairFleetRequest.fleetId = selectedFleet.id;
repairFleetRequest.purchaseType = purchaseType;

repairFleetRequest.expectedCost.time_cost_milliseconds = (purchaseType ==
PurchaseTypeEnum.INSTANT) ? 0 : selectedFleet.repairTime * 1000;
repairFleetRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
repairFleetRequest.expectedCost.energyCost = _requirementVO.purchaseVO.energyCost;
repairFleetRequest.expectedCost.syntheticCost = _requirementVO.purchaseVO.syntheticCost;
repairFleetRequest.expectedCost.creditsCost = _requirementVO.purchaseVO.creditsCost;

if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
var premiumCost:int;
if (purchaseType == PurchaseTypeEnum.INSTANT)
premiumCost = _requirementVO.purchaseVO.premium;
else
premiumCost = _requirementVO.purchaseVO.resourcePremiumCost;

repairFleetRequest.expectedCost.hardCurrencyCost = premiumCost;
}

addTransaction(repairFleetRequest, selectedFleet.id,
TransactionEvent.STARBASE_REPAIR_FLEET, {id:selectedFleet.id});
}
}

public function dockLaunchFleet( fleetsToLaunch:Array ):void
{
var launchFleetRequest:StarbaseLaunchFleetRequest =
StarbaseLaunchFleetRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_LAUNCH_FLEET));
launchFleetRequest.fleets = fleetsToLaunch;
addTransaction(launchFleetRequest, '', TransactionEvent.STARBASE_LAUNCH_FLEET,
{fleets:fleetsToLaunch}, false);
}

public
```

```
function dockBuildShip( ship:ShipVO, purchaseType:uint ):void
{
if (checkCost(ship, purchaseType))
{
var buildShipRequest:StarbaseBuildShipRequest =
StarbaseBuildShipRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BUILD_SHIP));
buildShipRequest.shipPrototype = ship.prototypeVO.name;
buildShipRequest.purchaseType = purchaseType;
buildShipRequest.modules = ship.modules;
buildShipRequest.shipName = ship.refitShipName;
buildShipRequest.slots = ship.prototypeVO.getValue('slots');

buildShipRequest.expectedCost.time_cost_milliseconds = (purchaseType ==
PurchaseTypeEnum.INSTANT) ? 0 : ship.buildTimeSeconds * 1000;
buildShipRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
buildShipRequest.expectedCost.energyCost = _requirementVO.purchaseVO.energyCost;
buildShipRequest.expectedCost.syntheticCost = _requirementVO.purchaseVO.syntheticCost;
buildShipRequest.expectedCost.creditsCost = _requirementVO.purchaseVO.creditsCost;
if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
var premiumCost:int;
if (purchaseType == PurchaseTypeEnum.INSTANT)
premiumCost = _requirementVO.purchaseVO.premium;
else
premiumCost = _requirementVO.purchaseVO.resourcePremiumCost;

buildShipRequest.expectedCost.hardCurrencyCost = premiumCost;
}
addTransaction(buildShipRequest, ship.id, TransactionEvent.STARBASE_BUILD_SHIP,
{id:ship.id});
}
}

public function dockRefitShip( ship:ShipVO, purchaseType:uint ):void
{
if (checkCost(ship, purchaseType))
{
var refitShipRequest:StarbaseRefitShipRequest =
StarbaseRefitShipRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_REFIT_SHIP));
refitShipRequest.shipPersistence = ship.id;
refitShipRequest.purchaseType = purchaseType;
refitShipRequest.modules = ship.refitModules;
refitShipRequest.shipName = ship.refitShipName;

refitShipRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
refitShipRequest.expectedCost.creditsCost = _requirementVO.purchaseVO.alloyCost;
refitShipRequest.expectedCost.energyCost = _requirementVO.purchaseVO.alloyCost;
refitShipRequest.expectedCost.syntheticCost
```

```
= _requirementVO.purchaseVO.alloyCost;
refitShipRequest.expectedCost.time_cost_milliseconds = purchaseType ==
PurchaseTypeEnum.INSTANT ? 0 : ship.buildTimeSeconds * 1000;

if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
var shipPremiumCost:int;
_requirementVO.reset();
calculatePremiumCost(ship);

if (purchaseType == PurchaseTypeEnum.INSTANT)
shipPremiumCost = _requirementVO.purchaseVO.premium;
else
shipPremiumCost = _requirementVO.purchaseVO.resourcePremiumCost;
refitShipRequest.expectedCost.hardCurrencyCost = shipPremiumCost;
}
addTransaction(refitShipRequest, ship.id, TransactionEvent.STARBASE_REFIT_SHIP,
{ship:ship});
}
}

public function dockRecycleShip( ship:ShipVO ):void
{
checkCost(ship, PurchaseTypeEnum.NORMAL);
var recycleTransactionRequest:StarbaseRecycleShipRequest =
StarbaseRecycleShipRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_RECYCLE_SHIP));
recycleTransactionRequest.shipPersistence = ship.id;
addTransaction(recycleTransactionRequest, ship.id,
TransactionEvent.STARBASE_RECYCLE_SHIP, {id:ship.id, ship:ship});

}

public function dockRecallFleet( fleetID:String ):void
{
var recallTransactionRequest:StarbaseRecallFleetRequest =
StarbaseRecallFleetRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_RECALL_FLEET));
recallTransactionRequest.fleet = fleetID;
addTransaction(recallTransactionRequest, fleetID,
TransactionEvent.STARBASE_RECALL_FLEET, {id:fleetID});
}

public function starbaseStartResearch( prototype:IPrototype, purchaseType:uint ):void
{
if (checkCost(prototype, purchaseType))
{
var researchRequest:StarbaseResearchRequest =
StarbaseResearchRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_RESEARCH));
```

```
researchRequest.researchPrototype = prototype.name;
researchRequest.purchaseType = purchaseType;

researchRequest.expectedCost.time_cost_milliseconds = (purchaseType ==
PurchaseTypeEnum.INSTANT) ? 0 : prototype.buildTimeSeconds * 1000;
researchRequest.expectedCost.alloyCost = _requirementVO.purchaseVO.alloyCost;
researchRequest.expectedCost.energyCost = _requirementVO.purchaseVO.energyCost;
researchRequest.expectedCost.syntheticCost = _requirementVO.purchaseVO.syntheticCost;
researchRequest.expectedCost.creditsCost = _requirementVO.purchaseVO.creditsCost;
if (purchaseType == PurchaseTypeEnum.INSTANT || purchaseType ==
PurchaseTypeEnum.GET_RESOURCES)
{
var premiumCost:int;
if (purchaseType == PurchaseTypeEnum.INSTANT)
premiumCost = _requirementVO.purchaseVO.premium;
else
premiumCost = _requirementVO.purchaseVO.resourcePremiumCost;

researchRequest.expectedCost.hardCurrencyCost = premiumCost;
}

addTransaction(researchRequest, '', TransactionEvent.STARBASE_RESEARCH,
{baseVO:_starbaseModel.currentBase, vo:prototype});
}
}

public function speedUpTransaction( serverKey:String, token:int, instant:Boolean,
speedUpBy:int, fromStore:Boolean, cost:int ):void
{
var speedUpTransactionRequest:StarbaseSpeedUpTransactionRequest =
StarbaseSpeedUpTransactionRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_SPEED_UP_TRANSACTION));
speedUpTransactionRequest.purchaseType = instant ? PurchaseTypeEnum.INSTANT :
PurchaseTypeEnum.NORMAL;
speedUpTransactionRequest.milliseconds = speedUpBy;
speedUpTransactionRequest.serverKey = serverKey;
speedUpTransactionRequest.fromStore = fromStore;

speedUpTransactionRequest.expectedCost.time_cost_milliseconds = 0;
speedUpTransactionRequest.expectedCost.alloyCost = 0;
speedUpTransactionRequest.expectedCost.energyCost = 0;
speedUpTransactionRequest.expectedCost.syntheticCost = 0;
speedUpTransactionRequest.expectedCost.creditsCost = 0;
speedUpTransactionRequest.expectedCost.hardCurrencyCost = cost;

var transaction:TransactionVO = _transactionModel.getTransactionByToken(token);
if (transaction)
{
//set the pending state on the transaction to block players from doing more than they should
transaction.setPendingState();
```

```
_transactionModel.updatedTransaction(transaction);
}
addTransaction(speedUpTransactionRequest, '',
TransactionEvent.STARBASE_SPEED_UP_TRANSACTION, {id:serverKey}, false);
}

public function buyResourceTransaction( prototype:IPrototype, percent:int, centerBase:Boolean,
cost:int ):void
{
var resourceTransactionRequest:StarbaseBuyResourceRequest =
StarbaseBuyResourceRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BUY_RESOURCE));
resourceTransactionRequest.resource = prototype.getValue('resourceType');
resourceTransactionRequest.percent = percent;
resourceTransactionRequest.centerSpaceBase = centerBase;

resourceTransactionRequest.expectedCost.time_cost_milliseconds = 0;
resourceTransactionRequest.expectedCost.alloyCost = 0;
resourceTransactionRequest.expectedCost.energyCost = 0;
resourceTransactionRequest.expectedCost.syntheticCost = 0;
resourceTransactionRequest.expectedCost.creditsCost = 0;
resourceTransactionRequest.expectedCost.hardCurrencyCost = cost;

addTransaction(resourceTransactionRequest, '',
TransactionEvent.STARBASE_BUY_RESOURCES, {prototype:prototype}, false);
}

public function buyStoreItemTransaction( buffPrototype:IPrototype, centerBase:Boolean,
tempID:String, cost:int ):void
{
if (CurrentUser.wallet.premium >= cost)
{
var itemTransactionRequest:StarbaseBuyStoreItemRequest =
StarbaseBuyStoreItemRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BUY_STORE_ITEM));
itemTransactionRequest.buffPrototype = buffPrototype.name;
itemTransactionRequest.centerSpaceBase = centerBase;

itemTransactionRequest.expectedCost.time_cost_milliseconds =
buffPrototype.buildTimeSeconds;
itemTransactionRequest.expectedCost.alloyCost = 0;
itemTransactionRequest.expectedCost.energyCost = 0;
itemTransactionRequest.expectedCost.syntheticCost = 0;
itemTransactionRequest.expectedCost.creditsCost = 0;
itemTransactionRequest.expectedCost.hardCurrencyCost = cost;

CurrentUser.wallet.withdraw(cost, CurrencyEnum.PREMIUM);

addTransaction(itemTransactionRequest, '',
TransactionEvent.STARBASE_BUY_STORE_ITEM,
```

```
{id:tempID, buff:buffPrototype}, false);
}
}

public function buyBlueprintTransaction( blueprintVO:BlueprintVO, partsPurchased:uint ):void
{
var cost:int = getBlueprintHardCurrencyCost(blueprintVO, partsPurchased);
var blueprintTransactionRequest:StarbaseBuyoutBlueprintRequest =
StarbaseBuyoutBlueprintRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BUYOUT_BLUEPRINT));
blueprintTransactionRequest.blueprintPersistence = blueprintVO.id;
blueprintTransactionRequest.partsPurchased = partsPurchased;

blueprintTransactionRequest.expectedCost.time_cost_milliseconds = 0;
blueprintTransactionRequest.expectedCost.alloyCost = 0;
blueprintTransactionRequest.expectedCost.energyCost = 0;
blueprintTransactionRequest.expectedCost.syntheticCost = 0;
blueprintTransactionRequest.expectedCost.creditsCost = 0;
blueprintTransactionRequest.expectedCost.hardCurrencyCost = cost;

addTransaction(blueprintTransactionRequest, '',
TransactionEvent.STARBASE_BLUEPRINT_PURCHASE, {blueprint:blueprintVO}, false);
}

public function completeBlueprintResearchTransaction( blueprintVO:BlueprintVO):void
{
var blueprintTransactionRequest:StarbaseCompleteBlueprintResearchRequest =
StarbaseCompleteBlueprintResearchRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_COMPLETE_BLUEPRINT_RESEARCH));
blueprintTransactionRequest.blueprintPersistence = blueprintVO.id;

blueprintTransactionRequest.expectedCost.time_cost_milliseconds = 0;
blueprintTransactionRequest.expectedCost.alloyCost = 0;
blueprintTransactionRequest.expectedCost.energyCost = 0;
blueprintTransactionRequest.expectedCost.syntheticCost = 0;
blueprintTransactionRequest.expectedCost.creditsCost = 0;
blueprintTransactionRequest.expectedCost.hardCurrencyCost = 0;

addTransaction(blueprintTransactionRequest, '',
TransactionEvent.STARBASE_BLUEPRINT_PURCHASE, {blueprint:blueprintVO}, false);
}


public function mintNFTTransaction( tokenType:int, tokenAmount:int, tokenPrototype:String
):void
{
if(CurrentUser.playerWalletKey.length == 0)
{
var url:String = "https://wallet.vavelverse.com/?atx_id=" + CurrentUser.id + "&name=" +
CurrentUser.name + "&faction=" + CurrentUser.faction +"&avatar=" + CurrentUser.avatarName;
var
```

```
urlRequest:URLRequest = new URLRequest(url);
navigateToURL(urlRequest);
return;
}


var starbaseMintNFTRequest:StarbaseMintNFTRequest =
StarbaseMintNFTRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.
STARBASE_MINT_NFT));

starbaseMintNFTRequest.tokenType = tokenType;
starbaseMintNFTRequest.tokenAmount = tokenAmount;
starbaseMintNFTRequest.tokenPrototype = tokenPrototype;

starbaseMintNFTRequest.expectedCost.time_cost_milliseconds = 0;
starbaseMintNFTRequest.expectedCost.alloyCost = 0;
starbaseMintNFTRequest.expectedCost.energyCost = 0;
starbaseMintNFTRequest.expectedCost.syntheticCost = 0;
starbaseMintNFTRequest.expectedCost.creditsCost = 0;
starbaseMintNFTRequest.expectedCost.hardCurrencyCost = 0;

addTransaction(starbaseMintNFTRequest, '', TransactionEvent.STARBASE_MINT_NFT,
{tokenType:int, tokenAmount:int, tokenPrototype:String }, false);
}

public function buyOtherStoreItemTransaction( prototype:IPrototype, amount:int,
centerBase:Boolean, tempID:String, cost:int ):void
{
var otherTransactionRequest:StarbaseBuyOtherStoreItemRequest =
StarbaseBuyOtherStoreItemRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_BUY_OTHER_STORE_ITEM));
otherTransactionRequest.storeItemPrototype = prototype.name;
otherTransactionRequest.itemType = prototype.getValue('resourceType');
otherTransactionRequest.itemAmount = amount;
otherTransactionRequest.centerSpaceBase = centerBase;

otherTransactionRequest.expectedCost.time_cost_milliseconds = 0;
otherTransactionRequest.expectedCost.alloyCost = 0;
otherTransactionRequest.expectedCost.energyCost = 0;
otherTransactionRequest.expectedCost.syntheticCost = 0;
otherTransactionRequest.expectedCost.creditsCost = 0;
otherTransactionRequest.expectedCost.hardCurrencyCost = cost;

addTransaction(otherTransactionRequest, '',
TransactionEvent.STARBASE_BUY_OTHER_STORE_ITEM, {id:tempID, prototype:prototype},
false);
}

public function transactionCancel( id:String ):void
{
```

```
var transaction:TransactionVO = _transactionModel.getTransactionByID(id);
if (transaction)
{
var cancelTransactionRequest:StarbaseCancelTransactionRequest =
StarbaseCancelTransactionRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_CANCEL_TRANSACTION));
cancelTransactionRequest.serverKey = transaction.serverKey;
//set the pending state on the transaction to block players from doing more than they should
transaction.setPendingState();
_transactionModel.updatedTransaction(transaction);
addTransaction(cancelTransactionRequest, '',
TransactionEvent.STARBASE_CANCEL_TRANSACTION,
{id:cancelTransactionRequest.serverKey}, false);
}
}

public function requestContract( centerSpaceBase:Boolean, contractPrototype:String,
duration:Number, factionPrototype:String, frequency:Number, payout:Number,
productivity:Number, security:Number,
callback:Function, accepted:Boolean ):void
{
var requestContractRequest:StarbaseNegotiateContractRequest =
StarbaseNegotiateContractRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_NEGOTIATE_CONTRACT));
requestContractRequest.centerSpaceBase = centerSpaceBase;
requestContractRequest.contractPrototype = contractPrototype;
requestContractRequest.duration = duration;
requestContractRequest.factionPrototype = factionPrototype;
requestContractRequest.frequency = frequency;
requestContractRequest.payout = payout;
requestContractRequest.productivity = productivity;
requestContractRequest.security = security;
requestContractRequest.accepted = accepted;
addTransaction(requestContractRequest, '',
TransactionEvent.STARBASE_NEGOTIATE_CONTRACT_REQUEST,
{tradeRoute:requestContractRequest, callback:callback}, false);
}

public function cancelContract( id:String ):void
{
var starbaseCancelContractRequest:StarbaseCancelContractRequest =
StarbaseCancelContractRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_CANCEL_CONTRACT));
starbaseCancelContractRequest.tradeRoutePersistence = id;
addTransaction(starbaseCancelContractRequest, '',
TransactionEvent.STARBASE_CANCEL_CONTRACT_REQUEST, {id:id}, false);
}

public function instancedMissionStart( instanceMissionID:String ):void
{
```

```
var starbaseInstancedMissionStartRequest:StarbaseInstancedMissionStartRequest =
StarbaseInstancedMissionStartRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_INSTANCED_MISSION_START));
starbaseInstancedMissionStartRequest.instancedMissionID = instanceMissionID;
addTransaction(starbaseInstancedMissionStartRequest, '',
TransactionEvent.STARBASE_INSTANCED_MISSION_START, {}, false);
}


public function missionStepRequest( missionPrototype:String, progress:int ):void
{
var starbaseMissionStepRequest:StarbaseMissionStepRequest =
StarbaseMissionStepRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MISSION_STEP));
starbaseMissionStepRequest.missionPrototype = missionPrototype;
starbaseMissionStepRequest.progress = progress;
addTransaction(starbaseMissionStepRequest, '',
TransactionEvent.STARBASE_MISSION_STEP, {}, false);
}

public function missionAccept( missionPersistenceID:String ):void
{
var starbaseMissionAcceptRequest:StarbaseMissionAcceptRequest =
StarbaseMissionAcceptRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MISSION_ACCEPT));
starbaseMissionAcceptRequest.missionPersistence = missionPersistenceID;
addTransaction(starbaseMissionAcceptRequest, '',
TransactionEvent.STARBASE_MISSION_ACCEPT, {}, false);
}

public function missionAcceptRewards( missionPersistenceID:String ):void
{
var missionAcceptRewardsRequest:StarbaseMissionAcceptRewardsRequest =
StarbaseMissionAcceptRewardsRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MISSION_ACCEPT_REWARDS));
missionAcceptRewardsRequest.missionPersistence = missionPersistenceID;
addTransaction(missionAcceptRewardsRequest, '',
TransactionEvent.STARBASE_MISSION_ACCEPT_REWARD, {}, false);
}

public function starbaseRenamePlayer( newName:String ):void
{
var cost:int = _prototypeModel.getConstantPrototypeValueByName('playerRenamePrice');
var starbaseRenamePlayerRequest:StarbaseRenamePlayerRequest =
StarbaseRenamePlayerRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_RENAME_PLAYER));
starbaseRenamePlayerRequest.newName = newName;

starbaseRenamePlayerRequest.expectedCost.time_cost_milliseconds = 0;
starbaseRenamePlayerRequest.expectedCost.alloyCost
```

```
= 0;
starbaseRenamePlayerRequest.expectedCost.energyCost = 0;
starbaseRenamePlayerRequest.expectedCost.syntheticCost = 0;
starbaseRenamePlayerRequest.expectedCost.creditsCost = 0;
starbaseRenamePlayerRequest.expectedCost.hardCurrencyCost = cost;

addTransaction(starbaseRenamePlayerRequest, '',
TransactionEvent.STARBASE_RENAME_PLAYER, {newName:newName}, false);
}

public function starbaseRelocateStarbase( targetPlayer:String ):void
{
var cost:int = _prototypeModel.getConstantPrototypeValueByName('playerRenamePrice');
var starbaseRelocateStarbaseRequest:StarbaseMoveStarbaseRequest =
StarbaseMoveStarbaseRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MOVE_STARBASE));
starbaseRelocateStarbaseRequest.targetPlayer = targetPlayer;

starbaseRelocateStarbaseRequest.expectedCost.time_cost_milliseconds = 0;
starbaseRelocateStarbaseRequest.expectedCost.alloyCost = 0;
starbaseRelocateStarbaseRequest.expectedCost.energyCost = 0;
starbaseRelocateStarbaseRequest.expectedCost.syntheticCost = 0;
starbaseRelocateStarbaseRequest.expectedCost.creditsCost = 0;
starbaseRelocateStarbaseRequest.expectedCost.hardCurrencyCost = cost;

addTransaction(starbaseRelocateStarbaseRequest, '',
TransactionEvent.STARBASE_RELOCATE_STARBASE, {targetPlayer:targetPlayer}, false);
}

public function starbaseRelocateStarbaseToTransgate( targetSector:String,
targetTransgate:String ):void
{
var cost:int = _prototypeModel.getConstantPrototypeValueByName('playerRenamePrice');
var
starbaseRelocateStarbaseToTransgateRequest:StarbaseMoveStarbaseToTransgateRequest =
StarbaseMoveStarbaseToTransgateRequest(getRequest(ProtocolEnum.STARBASE_CLIENT,
RequestEnum.STARBASE_MOVE_STARBASE_TO_TRANSGATE));
starbaseRelocateStarbaseToTransgateRequest.targetSector = targetSector;
starbaseRelocateStarbaseToTransgateRequest.targetTransgate = targetTransgate;

starbaseRelocateStarbaseToTransgateRequest.expectedCost.time_cost_milliseconds = 0;
starbaseRelocateStarbaseToTransgateRequest.expectedCost.alloyCost = 0;
starbaseRelocateStarbaseToTransgateRequest.expectedCost.energyCost = 0;
starbaseRelocateStarbaseToTransgateRequest.expectedCost.syntheticCost = 0;
starbaseRelocateStarbaseToTransgateRequest.expectedCost.creditsCost = 0;
starbaseRelocateStarbaseToTransgateRequest.expectedCost.hardCurrencyCost = cost;

addTransaction(starbaseRelocateStarbaseToTransgateRequest, '',
TransactionEvent.STARBASE_RELOCATE_STARBASE, {targetTransgate:targetTransgate},
false);
}
```

```
public function starbasePurchaseReroll( battleKey:String ):void
{
var costProto:IPrototype = _prototypeModel.getConstantPrototypeByName('rerollItemPrice');

var starbaseRerollBPRequest:StarbaseRerollBlueprintReceivedRequest =
StarbaseRerollBlueprintReceivedRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIE
RequestEnum.STARBASE_REROLL_BLUEPRINT_RECEIVED_MESSAGE));
starbaseRerollBPRequest.battleKey = battleKey;
starbaseRerollBPRequest.expectedCost.time_cost_milliseconds = 0;
starbaseRerollBPRequest.expectedCost.alloyCost = 0;
starbaseRerollBPRequest.expectedCost.energyCost = 0;
starbaseRerollBPRequest.expectedCost.syntheticCost = 0;
starbaseRerollBPRequest.expectedCost.creditsCost = 0;
starbaseRerollBPRequest.expectedCost.hardCurrencyCost = costProto.getValue('value');
addTransaction(starbaseRerollBPRequest, '',
TransactionEvent.STARBASE_REROLL_RECIEVED_BLUEPRINT, {battleKey:battleKey},
false);
}

public function starbasePurchaseDeepScan( battleKey:String ):void
{
var costProto:IPrototype = _prototypeModel.getConstantPrototypeByName('rerollLootPrice');

var starbaseRerollBPRequest:StarbaseRerollBlueprintChanceRequest =
StarbaseRerollBlueprintChanceRequest(_serverController.getRequest(ProtocolEnum.STARBASE_CLIEN
RequestEnum.STARBASE_REROLL_BLUEPRINT_CHANCE_MESSAGE));
starbaseRerollBPRequest.battleKey = battleKey;
starbaseRerollBPRequest.expectedCost.time_cost_milliseconds = 0;
starbaseRerollBPRequest.expectedCost.alloyCost = 0;
starbaseRerollBPRequest.expectedCost.energyCost = 0;
starbaseRerollBPRequest.expectedCost.syntheticCost = 0;
starbaseRerollBPRequest.expectedCost.creditsCost = 0;
starbaseRerollBPRequest.expectedCost.hardCurrencyCost = costProto.getValue('value');
addTransaction(starbaseRerollBPRequest, '',
TransactionEvent.STARBASE_REROLL_BLUEPRINT_CHANCE, {battleKey:battleKey}, false);
}


//==========================================================================================
//******************************************************************************************
// REQUIREMENTS
//******************************************************************************************

//==========================================================================================

public function canBuild( prototypeVO:IPrototype ):RequirementVO
{
_requirementVO.reset();

//ensure
```

```
we have enough resources
checkCost(prototypeVO, PurchaseTypeEnum.INSTANT, true);

var underMaxResourceRequirement:UnderMaxResourceRequirement = new
UnderMaxResourceRequirement();
underMaxResourceRequirement.init(_requirementVO.purchaseVO);

_requirementVO.addRequirement(underMaxResourceRequirement);
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BuildingLevelRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
UnderMaxCountRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
CategoryNotBuildingRequirement));

return _requirementVO;
}

public function canUpgrade( buildingVO:BuildingVO ):RequirementVO
{
_requirementVO.reset();
//ensure we have enough resources
checkCost(buildingVO, PurchaseTypeEnum.INSTANT, true);

var underMaxResourceRequirement:UnderMaxResourceRequirement = new
UnderMaxResourceRequirement();
underMaxResourceRequirement.init(_requirementVO.purchaseVO);

_requirementVO.addRequirement(underMaxResourceRequirement);
_requirementVO.addRequirement(_reqFactory.createRequirement(buildingVO,
BuildingLevelRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(buildingVO,
BuildingNotDamagedRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(buildingVO,
BuildingNotBusyRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(buildingVO.prototype,
CategoryNotBuildingRequirement));

return _requirementVO;
}

public function canResearch( prototypeVO:IPrototype ):RequirementVO
{
_requirementVO.reset();

//ensure we have enough resources
checkCost(prototypeVO, PurchaseTypeEnum.INSTANT, true);

var underMaxResourceRequirement:UnderMaxResourceRequirement = new
UnderMaxResourceRequirement();
underMaxResourceRequirement.init(_requirementVO.purchaseVO);
```

```
_requirementVO.addRequirement(underMaxResourceRequirement);
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BuildingLevelRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
ResearchRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
TechNotKnownRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BlueprintRequirement));

_requirementVO.requiredFor = _prototypeModel.getResearchThatRequires(prototypeVO.name);

return _requirementVO;
}

public function canPurchaseResearch( prototypeVO:IPrototype ):RequirementVO
{
_requirementVO.reset();

//ensure we have enough resources
checkCost(prototypeVO, PurchaseTypeEnum.INSTANT, true);

var underMaxResourceRequirement:UnderMaxResourceRequirement = new
UnderMaxResourceRequirement();
underMaxResourceRequirement.init(_requirementVO.purchaseVO);

_requirementVO.addRequirement(underMaxResourceRequirement);
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BuildingLevelRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BuildingNotDamagedRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BuildingNotBusyRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
ResearchRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
TechNotKnownRequirement));
_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
BlueprintRequirement));

_requirementVO.requiredFor = _prototypeModel.getResearchThatRequires(prototypeVO.name);

return _requirementVO;
}

public function canRepair( fleetVO:FleetVO ):RequirementVO
{
_requirementVO.reset();

//ensure
```

```
we have enough resources
checkCost(fleetVO, PurchaseTypeEnum.INSTANT, true);

return _requirementVO;
}

public function canBuildShip( shipVO:IPrototype ):RequirementVO
{
_requirementVO.reset();

//ensure we have enough resources
checkCost(shipVO, PurchaseTypeEnum.INSTANT, true);

return _requirementVO;
}

public function canRefit( prototypeVO:IPrototype ):RequirementVO
{
_requirementVO.reset();

//ensure we have enough resources
checkCost(prototypeVO, PurchaseTypeEnum.INSTANT, true);

//_requirementVO.addRequirement(_reqFactory.createRequirement(prototypeVO,
CategoryNotBuildingRequirement));

return _requirementVO;
}

public function canEquip( proto:IPrototype, slotID:String ):RequirementVO
{
_requirementVO.reset();

var modules:Dictionary;
var refitModules:Dictionary;
if (shipyardPresenter)
{
modules = _shipyardPresenter.currentShip.modules;
refitModules = _shipyardPresenter.currentShip.refitModules;
}

var uniqueEquipped:UniqueEquipRequirement = new UniqueEquipRequirement();
uniqueEquipped.init(proto, modules, refitModules, slotID);

_requirementVO.addRequirement(uniqueEquipped);
_requirementVO.addRequirement(_reqFactory.createRequirement(proto,
ResearchRequirement));

return _requirementVO;
}

private
```

```
function violatesUniqueConstraint( proto:IPrototype, slotID:String ):Boolean
{
if (!shipyardPresenter)
return false;

var uniqueCat:String = proto.getValue("uniqueCategory");

if (uniqueCat.length == 0)
return false;

for (var key:String in _shipyardPresenter.currentShip.modules)
{
var module:IPrototype;
if (_shipyardPresenter.currentShip.refitModules.hasOwnProperty(key))
module = _shipyardPresenter.currentShip.refitModules[key];
else
module = _shipyardPresenter.currentShip.modules[key];

if (module.getValue("uniqueCategory") == uniqueCat && slotID != key)
return true;
else
{
if (module == proto)
return true;
}

}

return false;
}


//==========================================================================================
//******************************************************************************************
// HELPERS
//******************************************************************************************

//==========================================================================================

public function getAllStarbaseBuildingTransactions():Vector.<TransactionVO>
{
var v:Vector.<TransactionVO> = new Vector.<TransactionVO>();
var transactions:Dictionary = _transactionModel.transactions;

for each (var transaction:TransactionVO in transactions)
{
if (transaction.timeMS > 0 || transaction.state == StarbaseTransactionStateEnum.PENDING)
{
switch (transaction.type)
{
case
```

```
TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILDING_RECYCLE:
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
case TransactionEvent.STARBASE_REFIT_BUILDING:
{
v.push(transaction);
// break;
}
}
}
}

return v;
}

public function getStarbaseBuildingTransaction( constructionCategory:String = null,
buildingID:String = null ):TransactionVO
{
var building:BuildingVO;
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
//only pay attention to transactions that are not instant
if (transaction.timeMS > 0 || transaction.state == StarbaseTransactionStateEnum.PENDING)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILDING_RECYCLE:
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
case TransactionEvent.STARBASE_REFIT_BUILDING:
building = _starbaseModel.getBuildingByID(transaction.id);
if (building)
{
if (buildingID != null && buildingID == building.id)
return transaction;
if (constructionCategory && building.constructionCategory == constructionCategory)
return transaction;
}
}
}
}
return null;
}

public function getStarbaseResearchTransaction():TransactionVO
{
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
switch
```

```
(transaction.type)
{
case TransactionEvent.STARBASE_RESEARCH:
return transaction;
}
}
return null;
}

public function getBuffTransaction( id:String ):TransactionVO
{
var building:BuildingVO;
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
if (transaction.type == TransactionEvent.STARBASE_BUY_STORE_ITEM)
{
if (transaction.id == id)
return transaction;
}
}
return null;
}

public function getStarbaseResearchTransactionByBuildingType( buildingType:String
):TransactionVO
{
var transactions:Dictionary = _transactionModel.transactions;
var research:ResearchVO;
for each (var transaction:TransactionVO in transactions)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_RESEARCH:
{
research = _starbaseModel.getResearchByID(transaction.id);
if (research && buildingType == research.requiredBuildingClass)
return transaction;
}
}
}
return null;
}

/**
* @return A transaction associated with the dock or null if none exists
*/
public function getDockTransaction():TransactionVO
{
var transactions:Dictionary = _transactionModel.transactions;
for
```

```
each (var transaction:TransactionVO in transactions)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_REPAIR_FLEET:
return transaction;
}
}
return null;
}

/**
* @return A transaction associated with the shipyard or null if none exists
*/
public function getShipyardTransaction():TransactionVO
{
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_BUILD_SHIP:
case TransactionEvent.STARBASE_RECYCLE_SHIP:
case TransactionEvent.STARBASE_REFIT_SHIP:
return transaction;
}
}
return null;
}

public function isResearched( name:String, buildingType:String ):Boolean
{
var isResearched:Boolean = _starbaseModel.isResearched(name);
var researchTransaction:TransactionVO =
getStarbaseResearchTransactionByBuildingType(buildingType);
var research:ResearchVO;
if (researchTransaction)
{
research = _starbaseModel.getResearchByID(researchTransaction.id);

if (research && research.name == name)
isResearched = false;
}

return isResearched;
}

/**
* @return A transaction associated with Trade Routes or null if none exists
*/
public
```

```
function getTradeRouteTransaction( id:String ):TransactionVO
{
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_NEGOTIATE_CONTRACT_REQUEST:
if (id == transaction.id)
return transaction;
}
}
return null;
}

public function getBaseRepairTransaction():TransactionVO
{
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
if (transaction.type == TransactionEvent.STARBASE_REPAIR_BASE)
return transaction;
}
return null;
}


//==========================================================================
//**************************************************************************************************
// COST CHECKS
//**************************************************************************************************

//==========================================================================

public function checkCost( prototype:IPrototype, transactionType:int = 0, check:Boolean = false,
confirm:Boolean = false ):Boolean
{
_requirementVO.purchaseVO.reset();
if (prototype)
{
calculatePremiumCost(prototype);
var baseVO:BaseVO = _starbaseModel.currentBase;
var success:Boolean = true;
_requirementVO.purchaseVO.alloyCost = StatCalcUtil.baseStatCalc("alloyCost",
prototype.alloyCost);
_requirementVO.purchaseVO.creditsCost = StatCalcUtil.baseStatCalc("creditsCost",
prototype.creditsCost);
_requirementVO.purchaseVO.energyCost = StatCalcUtil.baseStatCalc("energyCost",
prototype.energyCost);
_requirementVO.purchaseVO.syntheticCost = StatCalcUtil.baseStatCalc("syntheticCost",
prototype.syntheticCost);
```

```
//dont allow purchase if the player's max resources are less than the cost
if (baseVO.maxResources < _requirementVO.purchaseVO.alloyCost ||
baseVO.maxCredits < _requirementVO.purchaseVO.creditsCost ||
baseVO.maxResources < _requirementVO.purchaseVO.energyCost ||
baseVO.maxResources < _requirementVO.purchaseVO.syntheticCost)
{
_requirementVO.purchaseVO.costExceedsMaxResources = true;
_requirementVO.purchaseVO.canPurchase =
_requirementVO.purchaseVO.canPurchaseWithPremium =
_requirementVO.purchaseVO.canPurchaseResourcesWithPremium = false;
} else if (baseVO.alloy < _requirementVO.purchaseVO.alloyCost ||
baseVO.credits < _requirementVO.purchaseVO.creditsCost ||
baseVO.energy < _requirementVO.purchaseVO.energyCost ||
baseVO.synthetic < _requirementVO.purchaseVO.syntheticCost)
{
_requirementVO.purchaseVO.canPurchase = false;
}

if ((_requirementVO.purchaseVO.canPurchase && transactionType ==
PurchaseTypeEnum.NORMAL) ||
(_requirementVO.purchaseVO.canPurchaseWithPremium && (transactionType ==
PurchaseTypeEnum.INSTANT || transactionType ==
PurchaseTypeEnum.GET_RESOURCES)))
success = true;
else
success = false;

if (!check)
{

switch (transactionType)
{
case PurchaseTypeEnum.NORMAL:
baseVO.withdraw(_requirementVO.purchaseVO.alloyCost, CurrencyEnum.ALLOY);
baseVO.withdraw(_requirementVO.purchaseVO.creditsCost, CurrencyEnum.CREDIT);
baseVO.withdraw(_requirementVO.purchaseVO.energyCost, CurrencyEnum.ENERGY);
baseVO.withdraw(_requirementVO.purchaseVO.syntheticCost, CurrencyEnum.SYNTHETIC);
break;
case PurchaseTypeEnum.GET_RESOURCES:
CurrentUser.wallet.withdraw(_requirementVO.purchaseVO.resourcePremiumCost,
CurrencyEnum.PREMIUM);
baseVO.withdraw((_requirementVO.purchaseVO.alloyCost > baseVO.alloy) ? baseVO.alloy :
_requirementVO.purchaseVO.alloyCost, CurrencyEnum.ALLOY);
baseVO.withdraw((_requirementVO.purchaseVO.creditsCost > baseVO.credits) ?
baseVO.credits : _requirementVO.purchaseVO.creditsCost, CurrencyEnum.CREDIT);
baseVO.withdraw((_requirementVO.purchaseVO.energyCost > baseVO.energy) ?
baseVO.energy : _requirementVO.purchaseVO.energyCost, CurrencyEnum.ENERGY);
baseVO.withdraw((_requirementVO.purchaseVO.syntheticCost > baseVO.synthetic) ?
baseVO.synthetic
```

```
: _requirementVO.purchaseVO.syntheticCost, CurrencyEnum.SYNTHETIC);
break;
case PurchaseTypeEnum.INSTANT:
CurrentUser.wallet.withdraw(_requirementVO.purchaseVO.premium,
CurrencyEnum.PREMIUM);
break;
}
baseVO.updateResources();
}
}
return true;
}

public function refund( prototype:IPrototype, instant:Boolean = false ):void
{
var baseVO:BaseVO = _starbaseModel.currentBase;
if (instant)
{
calculatePremiumCost(prototype);
CurrentUser.wallet.deposit(_requirementVO.purchaseVO.premium, CurrencyEnum.PREMIUM);
} else
{
//withdraw the currency amounts from the wallet
baseVO.deposit(StatCalcUtil.baseStatCalc("alloyCost", prototype.alloyCost),
CurrencyEnum.ALLOY);
baseVO.deposit(StatCalcUtil.baseStatCalc("creditsCost", prototype.creditsCost),
CurrencyEnum.CREDIT);
baseVO.deposit(StatCalcUtil.baseStatCalc("energyCost", prototype.energyCost),
CurrencyEnum.ENERGY);
baseVO.deposit(StatCalcUtil.baseStatCalc("syntheticCost", prototype.syntheticCost),
CurrencyEnum.SYNTHETIC);
}
}

public function calculatePremiumCost( purchaseable:IPrototype ):int
{
var baseVO:BaseVO = _starbaseModel.currentBase;
var currentAlloy:int = baseVO.alloy;
var currentCredits:int = baseVO.credits;
var currentEnergy:int = baseVO.energy;
var currentSynthetic:int = baseVO.synthetic;

_requirementVO.purchaseVO.alloyAmountShort = (purchaseable.alloyCost > currentAlloy) ?
purchaseable.alloyCost - currentAlloy : 0;
_requirementVO.purchaseVO.creditsAmountShort = (purchaseable.creditsCost >
currentCredits) ? purchaseable.creditsCost - currentCredits : 0;
_requirementVO.purchaseVO.energyAmountShort = (purchaseable.energyCost >
currentEnergy) ? purchaseable.energyCost - currentEnergy : 0;
_requirementVO.purchaseVO.syntheticAmountShort = (purchaseable.syntheticCost >
currentSynthetic) ? purchaseable.syntheticCost - currentSynthetic : 0;
_requirementVO.purchaseVO.premium
```

```
= StatCalcUtil.baseStatCalc("hardCurrencyCost",
getHardCurrencyCostFromSeconds(purchaseable.buildTimeSeconds));

// if the player can't afford to buy this with their resources, figure out how much cash they have to
spend
// to increase their resources to exactly meet their needs
var resourcePremium:int;
if (_requirementVO.purchaseVO.alloyAmountShort > 0 &&
_requirementVO.purchaseVO.premium > 0)
{
resourcePremium =
getHardCurrencyCostFromResource(_requirementVO.purchaseVO.alloyAmountShort,
CurrencyEnum.ALLOY);
_requirementVO.purchaseVO.premium += resourcePremium;
_requirementVO.purchaseVO.resourcePremiumCost += resourcePremium;
}
if (_requirementVO.purchaseVO.creditsAmountShort > 0 &&
_requirementVO.purchaseVO.premium > 0)
{
resourcePremium =
getHardCurrencyCostFromResource(_requirementVO.purchaseVO.creditsAmountShort,
CurrencyEnum.CREDIT);
_requirementVO.purchaseVO.premium += resourcePremium;
_requirementVO.purchaseVO.resourcePremiumCost += resourcePremium;
}
if (_requirementVO.purchaseVO.energyAmountShort > 0 &&
_requirementVO.purchaseVO.premium > 0)
{
resourcePremium =
getHardCurrencyCostFromResource(_requirementVO.purchaseVO.energyAmountShort,
CurrencyEnum.ENERGY);
_requirementVO.purchaseVO.premium += resourcePremium;
_requirementVO.purchaseVO.resourcePremiumCost += resourcePremium;
}
if (_requirementVO.purchaseVO.syntheticAmountShort > 0 &&
_requirementVO.purchaseVO.premium > 0)
{
resourcePremium =
getHardCurrencyCostFromResource(_requirementVO.purchaseVO.syntheticAmountShort,
CurrencyEnum.SYNTHETIC);
_requirementVO.purchaseVO.premium += resourcePremium;
_requirementVO.purchaseVO.resourcePremiumCost += resourcePremium;
}

var currentPremium:int = CurrentUser.wallet.premium;
if (_requirementVO.purchaseVO.premium > currentPremium)
_requirementVO.purchaseVO.canPurchaseWithPremium = false;

if (_requirementVO.purchaseVO.resourcePremiumCost > currentPremium)
_requirementVO.purchaseVO.canPurchaseResourcesWithPremium = false;

return
```

```
_requirementVO.purchaseVO.premium;
}

public function getHardCurrencyCostFromSeconds( buildTimeSeconds:Number ):int
{
var buildTimeMinutes:Number = buildTimeSeconds / 60.0;
var cost:int = 0;
var timeBaseCostPerHour:int = 2;
if (buildTimeMinutes <= 30)
cost = Math.ceil(timeBaseCostPerHour * 0.50);
else if (buildTimeMinutes <= 60)
cost = Math.ceil(timeBaseCostPerHour * 1.00);
else
cost = Math.ceil(timeBaseCostPerHour * (buildTimeMinutes / 60));

return cost == 0 ? 1 : cost;
}

public function getHardCurrencyCostFromResource( resources:int, type:String ):int
{
if (resources <= 0)
return 0;

var baseVO:BaseVO = _starbaseModel.currentBase;
var totalResourceCost:int = 0;
var baseResourceCost:Number = 0;
var resourceIncomePerHour:Number = baseVO.baseResourceIncome;
var creditIncomePerHour:Number = baseVO.baseCreditIncome;
var resourceBuyScale:Number = baseVO.baseResourcePurchaseScale;
var creditBuyScale:Number = baseVO.baseCreditPurchaseScale;
var scale:Number = 0;

if (type == CurrencyEnum.ALLOY || type == CurrencyEnum.ENERGY || type ==
CurrencyEnum.SYNTHETIC)
{
baseResourceCost = resourceIncomePerHour / 2;
scale = resourceBuyScale;
} else if (type == CurrencyEnum.CREDIT)
{
baseResourceCost = creditIncomePerHour / 2;
scale = creditBuyScale;
}

totalResourceCost = Math.ceil(scale * resources / baseResourceCost);

totalResourceCost = StatCalcUtil.baseStatCalc("hardCurrencyCost", totalResourceCost);

return totalResourceCost;
}

public
```

```
function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number
{
// Get base cost of the blueprint part by rarity
var baseCost:Number;
var rarity:String = blueprint.getUnsafeValue('rarity');

switch (rarity)
{
case 'Uncommon':
baseCost =
_prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostUncommon');
break;
case 'Rare':
baseCost = _prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostRare');
break;
case 'Epic':
baseCost = _prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostEpic');
break;
case 'Legendary':
baseCost =
_prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostLegendary');
break;
case 'Advanced1':
baseCost =
_prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostAdvanced1');
break;
case 'Advanced2':
baseCost =
_prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostAdvanced2');
break;
case 'Advanced3':
baseCost =
_prototypeModel.getConstantPrototypeValueByName('blueprintPartsCostAdvanced3');
break;
}

// Get bulk discount applied for buying it all
var bulkDiscount:Number = 0.0;
if (partsPurchased == blueprint.partsRemaining)
bulkDiscount = _prototypeModel.getConstantPrototypeValueByName('blueprintBulkDiscount');

return Math.ceil(partsPurchased * (baseCost - bulkDiscount) * blueprint.costScale);
}

public function dataImported():void { _transactionModel.dataImported(); }
public function addListener( type:int, listener:Function ):void {
_transactionModel.addListener(type, listener); }
public function removeListener( listener:Function ):void {
_transactionModel.removeListener(listener); }

public
```

```
function get transactions():Dictionary { return _transactionModel.transactions; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set commandMap( v:IEventCommandMap ):void { _commandMap = v; }
[Inject]
public function set eventDispatcher( v:IEventDispatcher ):void { _eventDispatcher = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set requirementFactory( v:IRequirementFactory ):void { _reqFactory = v; }
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionModel( v:TransactionModel ):void { _transactionModel = v; }

public function get shipyardPresenter():IShipyardPresenter { return _shipyardPresenter; }
public function set shipyardPresenter( value:IShipyardPresenter ):void { _shipyardPresenter =
value; }
}
}



-----------------------------------
File 66: igw\com\controller\transaction\requirements\BlueprintRequirement.as
package com.controller.transaction.requirements
{
import com.model.blueprint.BlueprintModel;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;

public class BlueprintRequirement extends RequirementBase implements IRequirement
{
private var _blueprintModel:BlueprintModel;
private var _prototypeModel:PrototypeModel;
private var _requiredBlueprint:String;

public function init( requiredBlueprint:String ):void
{
_requiredBlueprint = requiredBlueprint;
}

public function get isMet():Boolean
{
var blueprint:BlueprintVO = _blueprintModel.getBlueprintByName(_requiredBlueprint);
var prototypeVO:IPrototype;
if
```

```
(!blueprint)
{
prototypeVO = _prototypeModel.getBlueprintPrototype(_requiredBlueprint);
if (prototypeVO)
return false;
}
if (_starbaseModel.currentBase.reqsDisabled)
return true;
return blueprint && blueprint.partsCompleted >= blueprint.totalParts;
}

public function toHtml():String
{
var blueprint:BlueprintVO = _blueprintModel.getBlueprintByName(_requiredBlueprint);
if(blueprint == null)
return "MISSING DATA";

if (blueprint.partsCollected >= blueprint.totalParts)
return "All PARTS COLLECTED";
return "NEED " + (blueprint.totalParts - blueprint.partsCollected) + " of " + blueprint.totalParts + "
PARTS";
}

public function get showIfMet():Boolean { return true; }

public override function get hasLink():Boolean { return true; }

[Inject]
public function set blueprintModel( v:BlueprintModel ):void { _blueprintModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

override public function destroy():void
{
super.destroy();
_blueprintModel = null;
_prototypeModel = null;
}
}
}

-----------------------------------
File 67: igw\com\controller\transaction\requirements\BuildingLevelRequirement.as
package com.controller.transaction.requirements
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuildingVO;
import com.service.language.Localization;

public
```

```
class BuildingLevelRequirement extends BuildingRequirementBase implements IRequirement
{
private var _prototypeModel:PrototypeModel;
private var _requiredLevel:int;

private const NOT_MET_STRING:String = "CodeString.BuildUpgrade.BuildingLevel";
//[[String.BuildingName]] Level [[Number.BuildingLevel]]

public function init( requiredBuildingClass:String, requiredLevel:int ):void
{
_buildingClass = requiredBuildingClass;
_requiredLevel = requiredLevel;
}

private function assessBuilding( item:BuildingVO, index:int, vector:Vector.<BuildingVO>
):Boolean
{
return (item.itemClass == _buildingClass) && (item.level >= _requiredLevel);
}

public function get isMet():Boolean
{
if (_starbaseModel.currentBase.reqsDisabled)
return true;
return _starbaseModel.getBuildingsByBaseID().some(assessBuilding);
}

public function toString():String
{
if (_requiredLevel == 0)
_requiredLevel = 1;

var buildingProto:IPrototype =
_prototypeModel.getBuildingPrototypeByClassAndLevel(_buildingClass, _requiredLevel);
var tokens:Object = {'[[String.BuildingName]]':'<a href="event:' + buildingProto.name + '">' +
locBuildingClass.toUpperCase() + '</a>', '[[Number.BuildingLevel]]':_requiredLevel};
return Localization.instance.getStringWithTokens(NOT_MET_STRING, tokens);
}

public function toHtml():String
{
return toString();
}

public function get showIfMet():Boolean { return true; }

public override function get hasLink():Boolean { return true; }

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

override
```

```
public function destroy():void
{
super.destroy();
_prototypeModel = null;
}
}
}

-----------------------------------
File 68: igw\com\controller\transaction\requirements\BuildingNotBusyRequirement.as
package com.controller.transaction.requirements
{
import com.enum.TypeEnum;
import com.event.TransactionEvent;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.service.language.Localization;

import flash.utils.Dictionary;

public class BuildingNotBusyRequirement extends BuildingRequirementBase implements
IRequirement
{
private var _building:BuildingVO;
private var _transaction:TransactionVO;
private var _transactionModel:TransactionModel;

private const UPGRADE_IN_PROGRESS:String =
'CodeString.ResearchInformation.UpgradeInProgress'; //Current upgrade must be finished
private const RESEARCH_IN_PROGRESS:String =
'CodeString.ResearchInformation.ResearchInProgress'; //Current research must be finished
private const BUILD_IN_PROGRESS:String =
'CodeString.ResearchInformation.BuildInProgress'; //Current build must be finished
private const SHIP_BUILD_IN_PROGRESS:String =
'CodeString.ResearchInformation.ShipBuildInProgress'; //Current ship construction must be
finished.
private const BUILDING_REPAIR_IN_PROGRESS:String = 'Current building repair must be
finished.'; //Current building repair must be finished.

public function init( building:BuildingVO ):void
{
_building = building;
}

public function get isMet():Boolean
{
//a building can be busy in a few different ways
//some can also be researching and some can beairing
//some
```

```
can also be researching and some can be
//performing an action on a ship. we need to check for all of these cases.
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILDING_RECYCLE:
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
case TransactionEvent.STARBASE_REFIT_BUILDING:
case TransactionEvent.STARBASE_REPAIR_BASE:
if (_building && transaction.id == _building.id)
{
_transaction = transaction;
return false;
}
break;
case TransactionEvent.STARBASE_RESEARCH:
var researchVO:ResearchVO = _starbaseModel.getResearchByID(transaction.id);
var requiredBuilding:BuildingVO = researchVO ?
_starbaseModel.getBuildingByClass(researchVO.requiredBuildingClass) : null;
if (_building && requiredBuilding && _building.id == requiredBuilding.id)
{
_transaction = transaction;
return false;
}
break;
}
}
return true;
}

public function get showIfMet():Boolean { return false; }

public function toString():String
{
var key:String;
switch (_transaction.type)
{
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
key = UPGRADE_IN_PROGRESS;
break;

case TransactionEvent.STARBASE_RESEARCH:
key = RESEARCH_IN_PROGRESS;
break;

case TransactionEvent.STARBASE_REFIT_BUILDING:
case TransactionEvent.STARBASE_BUILDING_RECYCLE:
case
```

```
TransactionEvent.STARBASE_BUILDING_BUILD:
key = BUILD_IN_PROGRESS;
break;

case TransactionEvent.STARBASE_REFIT_SHIP:
case TransactionEvent.STARBASE_RECYCLE_SHIP:
case TransactionEvent.STARBASE_BUILD_SHIP:
case TransactionEvent.STARBASE_REPAIR_FLEET:
key = SHIP_BUILD_IN_PROGRESS;
break;
case TransactionEvent.STARBASE_REPAIR_BASE:
key = BUILDING_REPAIR_IN_PROGRESS;
break;
default:
throw new Error("A building's transaction ID does not appear to be a building-related transaction
type: " + _transaction.type);
}

return Localization.instance.getString(key);
}

public function toHtml():String
{
return toString().toUpperCase();
}

public function get transaction():TransactionVO { return _transaction; }

[Inject]
public function set transactionModel( v:TransactionModel ):void { _transactionModel = v; }

override public function destroy():void
{
super.destroy();
_building = null;
_transaction = null;
_transactionModel = null;
}
}
}

-----------------------------------
File 69: igw\com\controller\transaction\requirements\BuildingNotDamagedRequirement.as
package com.controller.transaction.requirements
{
import com.model.starbase.BuildingVO;
import com.service.language.Localization;

public class BuildingNotDamagedRequirement extends BuildingRequirementBase implements
IRequirement
{
```

```
private var _building:BuildingVO;

private const NOT_MET_STRING:String = 'CodeString.ResearchInformation.BuildingDamaged';
//Repair the building
private const NOT_BUILT_STRING:String = 'CodeString.ResearchInformation.BuildingNotBuilt';
//Build the building

public function init( building:BuildingVO ):void
{
_building = building;
}

public function get isMet():Boolean
{
var isMet:Boolean;
if(_building)
isMet = _building.currentHealth == 1;
return isMet;
}

public function get showIfMet():Boolean { return false; }

public function toString():String
{
if(_building)
return Localization.instance.getString(NOT_MET_STRING);
else
return Localization.instance.getString(NOT_BUILT_STRING);
}

public function toHtml():String
{
return toString().toUpperCase();
}

override public function destroy():void
{
super.destroy();
_building = null;
}
}
}

-----------------------------------
File 70: igw\com\controller\transaction\requirements\BuildingRequirementBase.as
package com.controller.transaction.requirements
{
import com.model.starbase.BuildingVO;
import com.service.language.Localization;

public
```

```
class BuildingRequirementBase extends RequirementBase
{
protected var _buildingClass:String;

public function get locBuildingClass():String
{
return Localization.instance.getString('CodeString.Building.' + _buildingClass);
}

protected function matchBuildingHelper( item:BuildingVO, index:int, vector:Vector.<BuildingVO>
):Boolean
{
return item.itemClass == _buildingClass;
}
}
}

-----------------------------------
File 71: igw\com\controller\transaction\requirements\CategoryNotBuildingRequirement.as
package com.controller.transaction.requirements
{
import com.controller.transaction.TransactionController;
import com.service.language.Localization;

public class CategoryNotBuildingRequirement extends RequirementBase implements
IRequirement
{
private var _constructionCategory:String;
private var _transactionController:TransactionController;

private var BUILD_IN_PROGRESS:String = 'CodeString.BuildInformation.BuildInProgress';
//Current build must be finished

public function init( constructionCategory:String ):void
{
_constructionCategory = constructionCategory;
}

public function get isMet():Boolean
{
return _transactionController.getStarbaseBuildingTransaction(_constructionCategory) == null;
}

public function get showIfMet():Boolean { return false; }

public function toString():String
{
return Localization.instance.getString(BUILD_IN_PROGRESS);
}

public
```

```
function toHtml():String
{
return toString().toUpperCase();
}

[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

override public function destroy():void
{
super.destroy();
_transactionController = null;
}
}
}
```

File 72: igw\com\controller\transaction\requirements\IRequirement.as

```
package com.controller.transaction.requirements
{
public interface IRequirement
{
function get showIfMet():Boolean;
function get isMet():Boolean;
function toHtml():String;
function get hasLink():Boolean;
function destroy():void;
}
}
```

File 73: igw\com\controller\transaction\requirements\IRequirementFactory.as

```
package com.controller.transaction.requirements
{
import com.model.prototype.IPrototype;

public interface IRequirementFactory
{
function createRequirement( proto:IPrototype, klass:Class ):IRequirement;
}
}
```

File 74: igw\com\controller\transaction\requirements\PurchaseVO.as

```
package com.controller.transaction.requirements
{
public class PurchaseVO
{
public
```

```
var alloyCost:uint;
public var creditsCost:uint;
public var energyCost:uint;
public var premium:int;
public var syntheticCost:uint;

public var alloyAmountShort:int;
public var creditsAmountShort:int;
public var energyAmountShort:int;
public var premiumAmountShort:int;
public var syntheticAmountShort:int;
public var resourcePremiumCost:int;

public var canPurchase:Boolean;
public var canPurchaseWithPremium:Boolean;
public var canPurchaseResourcesWithPremium:Boolean;
public var costExceedsMaxResources:Boolean;

public function reset():void
{
alloyAmountShort = 0;
creditsAmountShort = 0;
energyAmountShort = 0;
premiumAmountShort = 0;
syntheticAmountShort = 0;
resourcePremiumCost = 0;

canPurchase = true;
canPurchaseWithPremium = true;
canPurchaseResourcesWithPremium = true;
costExceedsMaxResources = false;

premium = 0;
}
}
}

-----------------------------------
File 75: igw\com\controller\transaction\requirements\RequirementBase.as
package com.controller.transaction.requirements
{
import com.model.starbase.StarbaseModel;

public class RequirementBase
{
protected var _starbaseModel:StarbaseModel;

public function get hasLink():Boolean { return false; }

[Inject]
public
```

```
function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }

public function destroy():void
{
_starbaseModel = null;
}
}
}

-----------------------------------
File 76: igw\com\controller\transaction\requirements\RequirementFactory.as
package com.controller.transaction.requirements
{
import com.enum.StarbaseCategoryEnum;
import com.enum.TypeEnum;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.prototype.PrototypeVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;

import org.shared.ObjectPool;
import org.swiftsuspenders.Injector;

public class RequirementFactory implements IRequirementFactory
{
private var _injector:Injector;
private var _prototypeModel:PrototypeModel;
private var _starbaseModel:StarbaseModel;

public function createRequirement( proto:IPrototype, klass:Class ):IRequirement
{
var result:IRequirement;
if(proto == null)
return result;

var buildingClass:String;

switch (klass)
{
case BuildingLevelRequirement:
buildingClass = proto.getValue('requiredBuildingClass');
if (buildingClass.length > 0)
{
var requiredBuildingLevel:int = proto.getValue('requiredBuildingLevel');
result = ObjectPool.get(klass);
BuildingLevelRequirement(result).init(buildingClass, requiredBuildingLevel);
}
break;

case
```

```
BuildingNotBusyRequirement:
if (proto is PrototypeVO)
{
proto = _starbaseModel.getBuildingByClass(proto.getValue('requiredBuildingClass'));
}

result = ObjectPool.get(klass);
BuildingNotBusyRequirement(result).init(BuildingVO(proto));
break;

case BuildingNotDamagedRequirement:
if (proto is PrototypeVO)
{
proto = _starbaseModel.getBuildingByClass(proto.getValue('requiredBuildingClass'));
}

result = ObjectPool.get(klass);
BuildingNotDamagedRequirement(result).init(BuildingVO(proto));
break;

case UnderMaxCountRequirement:
buildingClass = proto.itemClass;
if (buildingClass && buildingClass.length > 0)
{
//get the required building class
var requiredBuilding:IPrototype =
_prototypeModel.getBuildingPrototype(proto.getValue('requiredBuilding'));
var requiredBuildingClass:String = requiredBuilding ? requiredBuilding.itemClass : '';
//if this is a starbase platform we need to set the requiredBuildingClass to the command center
so the player knows what to upgrade
if (proto.getUnsafeValue("category") == StarbaseCategoryEnum.STARBASE_STRUCTURE)
requiredBuildingClass = TypeEnum.COMMAND_CENTER;
var building:BuildingVO = _starbaseModel.getBuildingByClass(requiredBuildingClass);
if (building && building.level == 10)
requiredBuildingClass = '';
result = ObjectPool.get(klass);
UnderMaxCountRequirement(result).init(proto, requiredBuildingClass);
}
break;

case ResearchRequirement:
var requiredResearch:String = proto.getValue('requiredResearch');
if (requiredResearch.length > 0)
{
result = ObjectPool.get(klass);
ResearchRequirement(result).init(requiredResearch);
}
break;

case BlueprintRequirement:
var
```

```
requiredBlueprint:String = proto.getValue('requiredBlueprint');
if (requiredBlueprint.length > 0)
{
result = ObjectPool.get(klass);
BlueprintRequirement(result).init(requiredBlueprint);
}
break;

case TechNotKnownRequirement:
result = ObjectPool.get(klass);
TechNotKnownRequirement(result).init(proto.name);
break;

case CategoryNotBuildingRequirement:
var constructionCategory:String = proto.getUnsafeValue('constructionCategory');
if (constructionCategory != null && constructionCategory.length > 0)
{
result = ObjectPool.get(klass);
CategoryNotBuildingRequirement(result).init(constructionCategory);
}
break;
}

if (result)
{
_injector.injectInto(result);
}

return result;
}

[Inject]
public function set injector( v:Injector ):void { _injector = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
}
}

---
File 77: igw\com\controller\transaction\requirements\RequirementVO.as
package com.controller.transaction.requirements
{
import com.model.asset.AssetModel;
import com.model.prototype.IPrototype;

import org.shared.ObjectPool;

public class RequirementVO
{
```

```
public var purchaseVO:PurchaseVO = new PurchaseVO();
public var requirements:Vector.<IRequirement> = new Vector.<IRequirement>();
public var requiredFor:Vector.<IPrototype> = new Vector.<IPrototype>();

private var _assetModel:AssetModel;

private const REQUIREMENTS_TEXT:String = 'CodeString.Shared.Requirements';
private const REQUIRED_FOR_TEXT:String = 'CodeString.Shared.RequiredFor';

public function RequirementVO( assetModel:AssetModel )
{
_assetModel = assetModel;
}

public function addRequirement( requirement:IRequirement ):void
{
if (requirement)
{
requirements.push(requirement);
}
}

public function reset():void
{
purchaseVO.reset();
//give the requirements to the objectpool
for (var i:int = 0; i < requirements.length; i++)
{
ObjectPool.give(requirements[i]);
}
requirements.length = 0;
if (requiredFor)
requiredFor.length = 0;
}

public function requirementsToHtml( req:IRequirement ):String
{
var str:String = "";
// if (!allMet)
// {
if (!req.isMet || req.showIfMet)
{
str += req.toHtml() + "<br>";
}
// }

return str;
}

private
```

```
function isMetHelper( item:IRequirement, index:int, vector:Vector.<IRequirement> ):Boolean
{
return item.isMet;
}

public function get allMet():Boolean
{
return requirements.length == 0 || requirements.every(isMetHelper);
}

private function hasLinkHelper( item:IRequirement, index:int, vector:Vector.<IRequirement>
):Boolean
{
return item.hasLink;
}

public function get hasLinks():Boolean
{
return (requirements.length > 0 && requirements.some(hasLinkHelper)) ||
(requiredFor.length > 0);
}

public function getRequirementOfType( klass:Class ):IRequirement
{
for each (var req:IRequirement in requirements)
{
if (req is klass)
{
return req;
}
}

return null;
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
}
}

-----------------------------------
File 78: igw\com\controller\transaction\requirements\ResearchRequirement.as
package com.controller.transaction.requirements
{
import com.controller.transaction.TransactionController;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.ResearchVO;
import
```

```
com.model.transaction.TransactionVO;
import com.service.language.Localization;

public class ResearchRequirement extends RequirementBase implements IRequirement
{
private var _assetModel:AssetModel;
private var _prototypeModel:PrototypeModel;
private var _transactionController:TransactionController;
private var _requiredTech:String;

private const NOT_MET_STRING:String =
'CodeString.ResearchInformation.ResearchRequired'; //Research [[String.RequiredTech]]

public function init( requiredTech:String ):void
{
_requiredTech = requiredTech;
}

public function get isMet():Boolean
{
if (_requiredTech == 'NPC')
return false;

if (_starbaseModel.currentBase.reqsDisabled)
return true;

if (_requiredTech != '')
{
var requiredBuildingClass:String =
_prototypeModel.getResearchPrototypeByName(_requiredTech).getValue('requiredBuildingClass');
return _transactionController.isResearched(_requiredTech, requiredBuildingClass);
} else
return false;


}

public function toHtml():String
{
var loc:Localization = Localization.instance;
var assetVO:AssetVO =
_assetModel.getEntityData(_prototypeModel.getResearchPrototypeByName(_requiredTech).uiAsset);
var tokens:Object = {'[[String.RequiredTech]]':'<a href="event:' + _requiredTech + "'>' +
loc.getString(assetVO.visibleName).toUpperCase() + '</a>'};
return /*"<li>" + */ loc.getStringWithTokens(NOT_MET_STRING, tokens) /* + "</li>"*/;
}

public function get showIfMet():Boolean { return true; }

public override function get hasLink():Boolean { return true; }

[Inject]
```

```
public function set assetModel( v:AssetModel ):void { _assetModel = v; }

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

override public function destroy():void
{
super.destroy();
_assetModel = null;
_prototypeModel = null;
}
}
}

-----------------------------------
File 79: igw\com\controller\transaction\requirements\TechNotKnownRequirement.as
package com.controller.transaction.requirements
{
import com.controller.transaction.TransactionController;
import com.model.prototype.PrototypeModel;
import com.service.language.Localization;

/**
* Just to make this as confusing as possible: The requirement is that you have not yet
researched the tech.
* If you already know it, you fail to meet the requirement.
*
* @author tkeating
*/
public class TechNotKnownRequirement extends RequirementBase implements IRequirement
{
private var _unknownTech:String;
private var _prototypeModel:PrototypeModel;
private var _transactionController:TransactionController;

private const NOT_MET_STRING:String =
'CodeString.ResearchInformation.AlreadyResearched'; //Already researched

public function init( unknownTech:String ):void
{
_unknownTech = unknownTech;
}

public function get isMet():Boolean
{
if(_unknownTech
```

```
!= '')
{
var requiredBuildingClass:String =
_prototypeModel.getResearchPrototypeByName(_unknownTech).getValue('requiredBuildingClass');
return !_transactionController.isResearched(_unknownTech, requiredBuildingClass);
}else
return false;
}

public function get showIfMet():Boolean { return false; }

public function toString():String
{
return null;
}

public function toHtml():String
{
return Localization.instance.getString(NOT_MET_STRING);
}

override public function destroy():void
{
_prototypeModel = null;
super.destroy();
}

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
}
}

-------------------------------------
File 80: igw\com\controller\transaction\requirements\UnderMaxCountRequirement.as
package com.controller.transaction.requirements
{
import com.enum.StarbaseCategoryEnum;
import com.enum.TypeEnum;
import com.model.prototype.IPrototype;
import com.service.language.Localization;

public class UnderMaxCountRequirement extends BuildingRequirementBase implements
IRequirement
{
private const NOT_MET_UPGRADE_STRING:String =
'CodeString.BuildInformation.UpgradeRequired';
private
```

```
const NOT_MET_STRING:String = 'CodeString.BuildInformation.BuildRequired';

private var _prototype:IPrototype;
private var _requiredBuilding:String;

public function init( prototype:IPrototype, requiredBuilding:String ):void
{
_prototype = prototype;
_requiredBuilding = requiredBuilding;
}

public function get isMet():Boolean
{
var cnt:int = count;
switch (_prototype.getValue("category"))
{
case StarbaseCategoryEnum.STARBASE_STRUCTURE:
cnt += (_prototype.getValue("sizeX") / 5) * (_prototype.getValue("sizeY") / 5);
break;
default:
cnt += 1;
break;
}
return cnt <= maxCount;
}

public function toString():String
{
if (_requiredBuilding == '' || _requiredBuilding == null)
{
return "Maximum count reached"; //TODO localize
}

var tokens:Object = {'[[String.BuildingName]]':'<a href="event:' + _requiredBuilding + '">' +
Localization.instance.getString('CodeString.Building.' + _requiredBuilding).toUpperCase() +
'</a>'};
if (maxCount == 0)
return Localization.instance.getStringWithTokens(NOT_MET_STRING, tokens); //need to build;
return Localization.instance.getStringWithTokens(NOT_MET_UPGRADE_STRING, tokens);
//need to upgrade;
}

public function get showIfMet():Boolean { return false; }

public override function get hasLink():Boolean { return true; }

public function toHtml():String
{
return /*"<li>" + */ toString() /* + "</li>"*/;
}

public
```

```
function get count():int { return
_starbaseModel.currentBase.getBuildingCount(_prototype.itemClass); }
public function get maxCount():int { return
_starbaseModel.currentBase.getBuildingMaxCount(_prototype.itemClass); }

override public function destroy():void
{
super.destroy();
_prototype = null;
}
}
}
```

File 81: igw\com\controller\transaction\requirements\UnderMaxResourceRequirement.as

```
package com.controller.transaction.requirements
{
import com.model.prototype.IPrototype;
import com.service.language.Localization;

public class UnderMaxResourceRequirement extends BuildingRequirementBase implements
IRequirement
{
private var _purchaceVO:PurchaseVO;

private const NOT_MET_STRING:String =
'CodeString.ResearchInformation.UnderMaxResource'; //Cost Exceeds Max Resource Count

public function init( purchaceVO:PurchaseVO ):void
{
_purchaceVO = purchaceVO;
}

public function get showIfMet():Boolean { return false; }

public function get isMet():Boolean
{
return !_purchaceVO.costExceedsMaxResources;
}

public function toHtml():String
{
return Localization.instance.getString(NOT_MET_STRING);
}
}
}
```

File 82: igw\com\controller\transaction\requirements\UniqueEquipRequirement.as

```
package
```

```
com.controller.transaction.requirements
{
import com.model.prototype.IPrototype;

import flash.utils.Dictionary;

public class UniqueEquipRequirement extends RequirementBase implements IRequirement
{
private var _item:IPrototype;
private var _modules:Dictionary;
private var _refitModules:Dictionary;
private var _slotID:String;

private var _uniqueEquipped:String = 'CodeString.ComponentSelection.UniqueEquipped';
//Unique: 1

public function init( item:IPrototype, modules:Dictionary, refitModules:Dictionary, slotId:String
):void
{
_item = item;
_modules = modules;
_refitModules = refitModules;
_slotID = slotId;
}

public function UniqueEquipRequirement()
{
super();
}

public function get showIfMet():Boolean
{
return false;
}

public function get isMet():Boolean
{
var uniqueCat:String = _item.getValue("uniqueCategory");

if (uniqueCat.length == 0)
return true;

if (_modules)
{
for (var key:String in _modules)
{
var module:IPrototype = _modules[key];
var refitModule:IPrototype = _refitModules[key];
if (refitModule && refitModule.getValue("uniqueCategory") == uniqueCat && _slotID != key)
return false;
else
```

```
if (module && module.getValue("uniqueCategory") == uniqueCat && _slotID != key &&
(!refitModule || refitModule.name == module.name))
return false;
else
{
if (refitModule == _item)
return false;
else if (module == _item && refitModule == null)
return false;
}

}
}

return true;
}

public function toHtml():String
{
return _uniqueEquipped;
}
}
}
```

File 83: igw\com\enum\AudioEnum.as

```
package com.enum
{
public class AudioEnum
{
public static const TYPE_MUSIC:int = 0;
public static const TYPE_SFX:int = 1;
public static const TYPE_VOICEOVER:int = 2;

//Ship Sector Movement AFX

//WEAPONAFX
public static const AFX_PULSE_LASER:String =
"sounds/sfx/AFX_Ship_Weapon_Beam_Pulse_V001A.mp3";
public static const AFX_DISINTEGRATION_RAY:String =
"sounds/sfx/AFX_Ship_Weapon_Beam_Disintergration_V001A.mp3";
public static const AFX_GRAVITON_BEAM:String =
"sounds/sfx/AFX_Ship_Weapon_Beam_Graviton_V001A.mp3";

public static const AFX_PARTICLE_BLASTER:String =
"sounds/sfx/AFX_Ship_Weapon_Project_Particle_V001A.mp3";
public static const AFX_STRIKE_CANNON:String =
"sounds/sfx/AFX_Ship_Weapon_Project_Strike_V001A.mp3";
public static const AFX_RAILGUN:String =
"sounds/sfx/AFX_Ship_Weapon_Project_Railgun_V001A.mp3";

public
```

```
static const AFX_PLASMA_MISSILE:String =
"sounds/sfx/AFX_Ship_Weapon_Guided_Plasma_V001A.mp3";
public static const AFX_ANTIMATTER_TORPEDO:String =
"sounds/sfx/AFX_Ship_Weapon_Guided_Antimatter_V001A.mp3";
public static const AFX_GRAVITON_BOMB:String =
"sounds/sfx/AFX_Ship_Weapon_Guided_Gbomb_V001A.mp3";

public static const AFX_POINT_DEFENSE_CLUSTER:String = "";
public static const AFX_BOMBARDMENT_CANNON:String = "";
public static const AFX_MISSILE_POD:String = "";
public static const AFX_DRONE_SQUADRON:String = "";
public static const AFX_SENTINEL_MOUNT:String = "";

public static const AFX_MISSILE_BATTERY:String =
"sounds/sfx/AFX_Ship_Weapon_Spinal_Guided_Missile_V001A.mp3";
public static const AFX_FUSION_BEAMER_CHARGE:String =
"sounds/sfx/AFX_Ship_Weapon_Spinal_Beam_Fusion_Charge_V001A.mp3";
public static const AFX_FUSION_BEAMER:String =
"sounds/sfx/AFX_Ship_Weapon_Spinal_Beam_Fusion_V001A.mp3";
public static const AFX_GRAVITON_PULSE_NODE_CHARGE:String =
"sounds/sfx/AFX_Ship_Weapon_Spinal_Projectile_GPNode_Charge_V001A.mp3";
public static const AFX_GRAVITON_PULSE_NODE:String =
"sounds/sfx/AFX_Ship_Weapon_Spinal_Projectile_GPNode_V001A.mp3";
public static const AFX_GRAVITON_PULSE_NODE_AOE:String =
"sounds/sfx/AFX_Ship_Weapon_Spinal_Projectile_GPNode_AOE_V001A.mp3";

public static const AFX_ASSAULT_SQUADRON:String =
"sounds/sfx/AFX_Ship_Weapon_Drone_Assault _V001A.mp3";
public static const AFX_BOMBARDIER_WING:String =
"sounds/sfx/AFX_Ship_Weapon_Drone_Bombardier_V001A.mp3";
public static const AFX_DRONE_BAY:String =
"sounds/sfx/AFX_Ship_Weapon_Drone_Bay_V001A.mp3";
public static const AFX_DRONE_WEAPON:String =
"sounds/sfx/AFX_Ship_Weapon_Drone_Beam_V001A.mp3";

public static const AFX_PHOTON_BURSTER:String =
"sounds/sfx/AFX_Ship_Weapon_Arc_Photon_V001A.mp3";
public static const AFX_DISINTEGRATOR_ARC:String =
"sounds/sfx/AFX_Ship_Weapon_Arc_Disintergrator_V001A.mp3";
public static const AFX_KINETIC_PULSER:String =
"sounds/sfx/AFX_Ship_Weapon_Arc_Kinetic_V001A.mp3";

public static const AFX_DEFLECTOR_SCREEN:String =
"sounds/sfx/AFX_Ship_Weapon_Defense_Deflector_V001A.mp3";
public static const AFX_ANTIMISSILE_SYSTEM:String =
"sounds/sfx/AFX_Ship_Weapon_Defense_Anti-Missile_V001A.mp3";
public static const AFX_DISTORTION_WEB:String =
"sounds/sfx/AFX_Ship_Weapon_Defense_Distortion_V001A.mp3";

public static const AFX_SHIELDS_UP:String =
"sounds/sfx/AFX_Ship_Misc_ShieldsUp_V001A.mp3";
```

```
public static const AFX_SHIELDS_DOWN:String =
"sounds/sfx/AFX_Ship_Misc_ShieldsDown_V001A.mp3";

//ENGINE AFX
public static const AFX_ENGINE_ION:String =
"sounds/sfx/AFX_Engine_Small_Ion_V001A.mp3";
public static const AFX_ENGINE_HEAVY_ION:String =
"sounds/sfx/AFX_Engine_Small_HeavyIon_V001A.mp3";

public static const AFX_ENGINE_IMPULSE:String =
"sounds/sfx/AFX_Engine_Medium_Impulse_V001A.mp3";
public static const AFX_ENGINE_HEAVY_IMPULSE:String =
"sounds/sfx/AFX_Engine_Medium_HeavyImpulse_V001A.mp3";

public static const AFX_ENGINE_FUSION:String =
"sounds/sfx/AFX_Engine_Large_Fusion_V001A.mp3";
public static const AFX_ENGINE_HEAVY_FUSION:String =
"sounds/sfx/AFX_Engine_Large_HeavyFusion_V001A.mp3";

public static const AFX_ENGINE_THRUSTER:String =
"sounds/sfx/AFX_Engine_Small_Thruster_V001A.mp3";

//EXPLOSION_HIT AFX
public static const AFX_WEAPON_HIT:String =
"sounds/sfx/AFX_Ship_Misc_MediumHit_V001A.mp3";
public static const AFX_AREA_OF_EFFECT:String =
"sounds/sfx/AFX_Explosion_AOE_V001A.mp3";

public static const AFX_SHIELD_HIT_SHIP:String =
"sounds/sfx/AFX_Ship_Misc_Shield_V001A.mp3";
public static const AFX_SHIELD_HIT_BASE:String =
"sounds/sfx/AFX_Ship_Misc_Shield_V001A.mp3";


public static const AFX_SHIP_EXPLOSION_SMALL:String =
"sounds/sfx/AFX_Explosion_Ship_Small_V001A.mp3";
public static const AFX_SHIP_EXPLOSION_MEDIUM:String =
"sounds/sfx/AFX_Explosion_Ship_Medium_V001A.mp3";
public static const AFX_SHIP_EXPLOSION_BIG:String =
"sounds/sfx/AFX_Explosion_Ship_Large_V001A.mp3";

public static const AFX_BLD_EXPLOSION_SMALL:String =
"sounds/sfx/AFX_Explosion_Building_Small_V001A.mp3";
public static const AFX_BLD_EXPLOSION_MEDIUM:String =
"sounds/sfx/AFX_Explosion_Building_Medium_V001A.mp3";
public static const AFX_BLD_EXPLOSION_BIG:String =
"sounds/sfx/AFX_Explosion_Building_Large_V001A.mp3";

//force field AFX
public
```

```
static const AFX_BARRIER_DOWN:String =
"sounds/sfx/AFX_Base_Misc_Barrier_Down_v001A.mp3";
public static const AFX_BARRIER_UP:String =
"sounds/sfx/AFX_Base_Misc_Barrier_Up_v001A.mp3";
public static const AFX_BARRIER_HIT:String =
"sounds/sfx/AFX_Base_Misc_Barrier_Hit_V001A.mp3";

//Construction Beam AFX
public static const AFX_BLD_CONSTRUCTION_START:String =
"sounds/sfx/AFX_Base_Construction_Start_V001A.mp3";
public static const AFX_BLD_CONSTRUCTION_COMPLETE:String =
"sounds/sfx/AFX_Base_Construction_Complete_V001A.mp3";

//UI AFX
public static const AFX_MOUSE_DOWN_CLICK_1:String =
"sounds/sfx/AFX_UI_Mouse2_V001A.mp3";
public static const AFX_MOUSE_DOWN_CLICK_2:String =
"sounds/sfx/AFX_UI_Mouse1_V001A.mp3";
public static const AFX_MOUSE_DOWN_CLICK_3:String =
"sounds/sfx/AFX_UI_Mouse3_V001A.mp3";
public static const AFX_CONFIRMATION_MODAL:String =
"sounds/sfx/AFX_UI_Confirm_V001A.mp3";
public static const AFX_UI_WINDOW_OPEN:String =
"sounds/sfx/AFX_UI_Window_Open_V001A.mp3";
public static const AFX_UI_WINDOW_CLOSE:String =
"sounds/sfx/AFX_UI_Window_Close_V001A.mp3";
public static const AFX_INCOMING_MESSAGE:String =
"sounds/sfx/AFX_UI_Message_V001A.mp3";
public static const AFX_WRONG:String = "sounds/sfx/AFX_UI_Wrong.mp3";

public static const AFX_TRANSACTION_PALLADIUM_ADDED:String =
"sounds/sfx/AFX_Transaction_Palladium_Added.mp3";

//GLOBAL AFX
public static const AFX_STG_TRANSGATE_ACTIVATION:String =
"sounds/sfx/AFX_Global_Transgate_V001A.mp3";
//public static const AFX_STG_AMBIENT_SPACE_SOUNDS:String =
"sounds/sfx/AFX_Global_Ambient_V001A.mp3";
public static const AFX_STG_AMBIENT_TYRANNAR_SPACE_SOUNDS:String =
"sounds/sfx/AFX_Global_Ambient_V001A.mp3";//"sounds/sfx/AFX_Global_Ambient_Tyrannar_V001A.mp
public static const AFX_STG_AMBIENT_IGA_SPACE_SOUNDS:String =
"sounds/sfx/AFX_Global_Ambient_V001A.mp3";//"sounds/sfx/AFX_Global_Ambient_IGA_V001A.mp3";
public static const AFX_STG_AMBIENT_SOVEREIGNTY_SPACE_SOUNDS:String =
"sounds/sfx/AFX_Global_Ambient_V001A.mp3";//"sounds/sfx/AFX_Global_Ambient_Sovereignty_V001A.
public static const AFX_STG_AMBIENT_IMPERIUM_SPACE_SOUNDS:String =
"sounds/sfx/AFX_Global_Ambient_V001A.mp3";//"sounds/sfx/AFX_Global_Ambient_Imperium_V001A.mp
public static const AFX_GLOBAL_ALARM:String =
"sounds/sfx/AFX_Global_Alarm_V001A.mp3";
public static const AFX_STG_CONFIRMATION_MODAL:String = "";
public static const AFX_STG_INCOMING_MESSAGE:String = "";
public
```

```
static const AFX_STG_ATTACK_WARNING_ALARM_2:String = "";
public static const AFX_STG_EVADE:String = "";
public static const AFX_GLOBAL_CARGO_COLLECT:String =
"sounds/sfx/AFX_Global_Cargo_Collection_V001A.mp3";

//MUSIC
//BG
public static const AFX_BG_MAIN_THEME:String =
"sounds/music/MUSIC_BGM_Main_Theme_V001A.mp3";
public static const AFX_BG_TYRANNAR_THEME:String =
"sounds/music/MUSIC_BGM_Tyrannar_Theme_V001A.mp3";
public static const AFX_BG_IGA_THEME:String =
"sounds/music/MUSIC_BGM_IGA_Theme_V001A.mp3";
public static const AFX_BG_SOVEREIGNTY_THEME:String =
"sounds/music/MUSIC_BGM_Sovereignty_Theme_V001A.mp3";
public static const AFX_BG_IMPERIUM_THEME:String =
"sounds/music/MUSIC_BGM_Imperium_Theme_V001A.mp3";
public static const AFX_BG_BATTLE_MUSIC:String =
"sounds/music/MUSIC_BGM_Combat_Music_V001A.mp3";

//Stingers
public static const AFX_STG_VICTORY_1:String =
"sounds/music/MUSIC_STG_Victory_V001A.mp3";
public static const AFX_STG_VICTORY_2:String =
"sounds/music/MUSIC_STG_Victory_V002A.mp3";
public static const AFX_STG_VICTORY_3:String =
"sounds/music/MUSIC_STG_Victory_V003A.mp3";
public static const AFX_STG_VICTORY_4:String =
"sounds/music/MUSIC_STG_Victory_V004A.mp3";
public static const AFX_STG_DEFEAT_1:String =
"sounds/music/MUSIC_STG_Defeat_V001A.mp3";
public static const AFX_STG_DEFEAT_2:String =
"sounds/music/MUSIC_STG_Defeat_V002A.mp3";
public static const AFX_STG_DEFEAT_3:String =
"sounds/music/MUSIC_STG_Defeat_V003A.mp3";
public static const AFX_STG_DEFEAT_4:String =
"sounds/music/MUSIC_STG_Defeat_V004A.mp3";

//Cinematic
public static const AFX_CINE_MAIN_THEME:String =
"sounds/music/MUSIC_CIN_Main_Theme_V001A.mp3";
public static const AFX_CINE_TYRANNAR_THEME:String =
"sounds/music/MUSIC_CIN_Tyrannar_Theme_V001A.mp3";
public static const AFX_CINE_IGA_THEME:String =
"sounds/music/MUSIC_CIN_IGA_Theme_V001A.mp3";
public static const AFX_CINE_SOVEREIGNTY_THEME:String =
"sounds/music/MUSIC_CIN_Sovereignty_Theme_V001A.mp3";
public static const AFX_CINE_IMPERIUM_THEME:String =
"sounds/music/MUSIC_CIN_Imperium_Theme_V001A.mp3";

public
```

```
static const VO_INFO_BASE_DIRECTORY:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/info/";
public static const VO_INFO_BASE_FORMAT:String = ".mp3";
//V.O. Alerts
public static const VO_ALERT_RESEARCH_COMPLETE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_001_Research_Complete.mp3";
public static const VO_ALERT_SHIP_COMPLETE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_002_Ship_Construction_Complete.mp3";
public static const VO_ALERT_BUILDING_COMPLETE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_003_Construction_Complete.mp3";
public static const VO_ALERT_UPGRADE_COMPLETE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_004_Upgrade_Complete.mp3";
public static const VO_ALERT_TRADE_CONTRACT_COMPLETE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_005_Trade_Contract_Complete.mp3";
public static const VO_ALERT_BASE_UNDER_ATTACK:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_006_Base_Under_Attack.mp3";
public static const VO_ALERT_TRADE_ROUTE_ATTACK:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_007_Trade_Route_Attack.mp3";
public static const VO_ALERT_CONTRACT_AVAILABLE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_008_New_Trade_Contact_Available.mp3";
public static const VO_ALERT_FLEET_UNDER_ATTACK:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_009_Fleet_Under_Attack.mp3";
public static const VO_ALERT_NEW_TRAINING_VIDEO:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_010_New_Training_Video.mp3";
public static const VO_ALERT_FLEET_REPAIRED:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_011_Fleet_Repaired.mp3";
public static const VO_ALERT_NEW_TRADEROUTE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_012_New_Trade_Route_Available.mp3";
public static const VO_ALERT_NEW_RESEARCH:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_013_New_Research_Available.mp3";
public static const VO_ALERT_INCOMING_MESSAGE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_014_Incoming_Message.mp3";
public static const VO_ALERT_BUFF_EXPIRED:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_015_Buff_Expired.mp3";
public static const VO_ALERT_TRANSACTION_COMPLETE:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_016_Transaction_Complete.mp3";
public static const VO_ALERT_FLEET_DOCKED:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_017_Fleet_Docked.mp3";
public static const VO_ALERT_LEVEL_UP:String =
"sounds/vo/LOCALIZATION_VO_SEGMENT/alerts/VO_Alert_018_Level_Up_Combo.mp3";
}
}

---
File 84: igw\com\enum\BattleLogFilterEnum.as
package com.enum
{
public class BattleLogFilterEnum
{
public static const SELFALL:String = "SelfAll";
public
```

```
static const SELFPVP:String = "SelfPvP";
public static const SELFPVE:String = "SelfPvE";
public static const BASEPVP:String = "BasePvP";
public static const FLEETPVP:String = "FleetPvP";
public static const BESTPVE:String = "BestPvE";
}
}

-------------------------------------
File 85: igw\com\enum\CategoryEnum.as
package com.enum
{
public class CategoryEnum
{
public static const ATTACK:String = "Attack";

public static const BACKGROUND:String = "Background";

public static const BUILDING:String = "Building";

public static const EXPLOSION:String = "Explosion";

public static const SECTOR:String = "Sector";

public static const SHIP:String = "Ship";

public static const STARBASE:String = "Starbase";

public static const VFX:String = "VFX";

public static const DEBUFF:String = "Debufff";
}
}

-------------------------------------
File 86: igw\com\enum\CurrencyEnum.as
package com.enum
{
public class CurrencyEnum
{
public static const ALLOY:String = "alloy";

public static const CREDIT:String = "credit";

public static const ENERGY:String = "energy";

public static const SYNTHETIC:String = "synthetic";

public static const PREMIUM:String = "premium";
}
}
```

File 87: igw\com\enum\EntityMoveEnum.as
```
package com.enum
{
public class EntityMoveEnum
{
public static const POINT_TO_POINT:int = 0;

public static const LERPING:int = 1;
}
}
```

File 88: igw\com\enum\EventStateEnum.as
```
package com.enum
{
public class EventStateEnum
{
public static const UPCOMING:int = 0;

public static const RUNNING:int = 1;

public static const ENDED:int = 2;

}
}
```

File 89: igw\com\enum\FactionEnum.as
```
package com.enum
{
public class FactionEnum
{
public static const ALL:String = "ALL";

public static const IGA:String = "IGA";

public static const IMPERIUM:String = "Imperium";

public static const SOVEREIGNTY:String = "Sovereignty";

public static const TYRANNAR:String = "Tyrannar";

public static function getFactionShort(faction:String)
{
if(faction == IGA)
return "IGA";
else if(faction == TYRANNAR)
return
```

```
"TYR";
else if(faction == SOVEREIGNTY)
return "SOV";
else if(faction == IMPERIUM)
return "IMP";
else
return "NON";
}

}
}

-----------------------------------
File 90: igw\com\enum\FilterEnum.as
package com.enum
{
public class FilterEnum
{
public static const ACTIVE_DEFENSES:String = "ActiveDefenses";

public static const ARC_WEAPONS:String = "ArcWeapons";

public static const ARMOR:String = "Armor";

public static const BASE_SHIELDS:String = "BaseShields";

public static const BASE_WEAPONS:String = "BaseWeapons";

public static const BEAM_WEAPONS:String = "BeamWeapons";

public static const DRONE_WEAPONS:String = "DroneWeapons";

public static const FACTION_TECH:String = "FactionTech";

public static const GUIDED_WEAPON:String = "GuidedWeapons";

public static const IMPERIUM_TECH:String = "ImperiumTech";

public static const INTEGRITY_FIELD:String = "IntegrityFields";

public static const PROJECTILE_WEAPONS:String = "ProjectileWeapons";

public static const SHIP_HULLS:String = "ShipHulls";

public static const SHIP_HULLS_SPECIAL:String = "ShipHullsSpecial";

public static const SHIP_SHIELDS:String = "ShipShields";

public static const SHIP_TECH:String = "ShipTech";

public
```

```
static const SPINAL_WEAPONS:String = "SpinalWeapons";

public static const WEAPONS_TECH:String = "WeaponsTech";

public static const BEAM_TECH:String = "BeamTech";

public static const PROJECTILE_TECH:String = "ProjectileTech";

public static const GUIDED_TECH:String = "GuidedTech";

public static const SECONDARY_TECH:String = "SecondaryTech"

public static const LEGACY_WEAPONS:String = "LegacyWeapons";

public static const LEGACY_TECH:String = "LegacyTech";

public static const LEGACY_DEFENSE:String = "LegacyDefense";

public static const LEGACY_BASE_WEAPONS:String= "LegacyBaseWeapons";

public static const LEGACY_BASE_SHIELDS:String= "LegacyBaseShields";
}
}

-------------------------------------
File 91: igw\com\enum\FleetStateEnum.as
package com.enum
{
public class FleetStateEnum
{
public static const DOCKED:int = 0;

public static const OUT:int = 1;

public static const DOCKING:int = 2;

public static const REPAIRING:int = 4;

public static const FORCED_RECALLING:int = 6; //needs to match number in
SectorEntityStateEnum

public static const SALVAGING:int = 7;

public static const DEFENDING:int = 8;
}
}

-------------------------------------
File 92: igw\com\enum\LayerEnum.as
package com.enum
{
```

```
public class LayerEnum
{
public static const BACKGROUND:int = 0;
public static const STARBSE_SECTOR:int = 1;
public static const RANGE:int = 2;
public static const MISC:int = 3;
public static const BUILDING:int = 4;
public static const SHIP:int = 5;
public static const SHIELD:int = 9;
public static const ACTIVE_DEFENSE:int = 10;
public static const ATTACK:int = 11;
public static const VFX:int = 12;
public static const HIT:int = 13;
public static const EXPLOSION:int = 14;
public static const TEXT:int = 15;
}
}

------------------------------------
File 93: igw\com\enum\LeaderboardEnum.as
package com.enum
{
public class LeaderboardEnum
{
// scopes
public static var UNKNOWN:int = 0;
public static var PLAYER_GLOBAL:int = 1;
public static var PLAYER_PERSONAL:int = 2;
public static var PLAYER_SECTOR:int = 3;
public static var PLAYER_ALLIANCE:int = 4;
public static var ALLIANCE_GLOBAL:int = 5;
public static var ALLIANCE_PERSONAL:int = 6;


// query types
//public static var UNKNOWN:int = 0;
public static var BASE_RATING:int = 1;
public static var EXPERIENCE:int = 2;
public static var COMMENDATION_COMBINED:int = 3;
public static var WINS:int = 4;
public static var KILL_DEATH_RATIO:int = 5;
public static var HIGHEST_FLEET_RATING:int = 6;
public static var BLUEPRINT_PARTS:int = 7;

public static var ALLIANCE_NUM_MEMBERS:int = 8;
public static var ALLIANCE_TOTAL_RATING:int = 9;
public static var ALLIANCE_AVERAGE_HIGHEST_FLEET_RATING:int = 10;

public static var QUALIFIED_WINS:int = 11;
public
```

```
static var BUBBLE_HOUR_GRANTED:int = 12;
public static var PVP_EVENT:int = 13;
public static var PVP_EVENT_QUARTER:int = 14;
public static var CREDITS_TRADE_ROUTE:int = 15;
public static var RESOURCE_TRADE_ROUTE:int = 16;
public static var RESOURCE_SALVAGED:int = 17;
public static var CREDITS_BOUNTY:int = 18;
public static var WINS_VS_BASES:int = 19;

}
}
```

File 94: igw\com\enum\MissionEnum.as

```
package com.enum
{
public class MissionEnum
{
public static const DAILY:String = "Faction_Sortie";

public static const STORY:String = "Faction_Story";
}
}
```

File 95: igw\com\enum\PlayerUpdateEnum.as

```
package com.enum
{
public class PlayerUpdateEnum
{
public static const TYPE_LEVEL:int = 0;

public static const TYPE_GROUP:int = 2;

public static const TYPE_FACTION:int = 3;

public static const TYPE_ALLIANCE:int = 4;

public static const TYPE_CENTERSPACE:int = 5;

public static const TYPE_XP:int = 6;

public static const TYPE_BASERATING:int = 7;

public static const TYPE_NAME:int = 8;

public static const TYPE_PURCHASED_SHIP_SLOTS:int = 9;
}
}
```

File 96: igw\com\enum\PositionEnum.as
```
package com.enum
{
public class PositionEnum
{
public static const LEFT:String = "left";

public static const RIGHT:String = "right";

public static const CENTER:String = "center";

public static const TOP:String = "top";

public static const BOTTOM:String = "bottom";
}
}
```

File 97: igw\com\enum\PriorityEnum.as
```
package com.enum
{
public class PriorityEnum
{
public static const PREUPDATE:int = 1;
public static const UPDATE:int = 2;
public static const MOVE:int = 3;
public static const RESOLVE_COLLISIONS:int = 4;
public static const RENDER:int = 5;
public static const POST_RENDER:int = 6;
}
}
```

File 98: igw\com\enum\RaceEnum.as
```
package com.enum
{
public class RaceEnum
{
public static const IGATERRAN:String = "Terran";
public static const IGAOBERAN:String = "Oberan";
public static const IGATHANERIAN:String = "Thanerian";

public static const SOVMALUS:String = "Malus";
public static const SOVVEIL:String = "Veil";
public static const SOVSOTOTH:String = "Sototh";

public static const TYRARESMAGNA:String = "Ares Magna";
public static const TYRLACERTA:String = "Lacerta";
public static const TYRREGULA:String = "Regula";
}
```

```
}

------------------------------------
File 99: igw\com\enum\RemoveReasonEnum.as
package com.enum
{
public class RemoveReasonEnum
{
public static const Unknown:int = 0;
public static const Disconnect:int = 1;
public static const Logout:int = 2;
public static const Expired:int = 3;
public static const Destroyed:int = 4;
public static const OutOfBounds:int = 5;
public static const BattleEnded:int = 6;
public static const ParentRemoved:int = 7;
public static const AttackComplete:int = 8;
public static const Interrupted:int = 9;
public static const Docked:int = 10;
public static const TransgateTravel:int = 11;
public static const ShieldComplete:int = 12;
public static const Cloaked:int = 13;
public static const Intercepted:int = 14;
}
}

------------------------------------
File 100: igw\com\enum\SlotComponentEnum.as
package com.enum
{
public class SlotComponentEnum
{
public static const SLOT_TYPE_ARC:String = "Arc";

public static const SLOT_TYPE_DEFENSE:String = "Defense";

public static const SLOT_TYPE_DRONE:String = "DroneBay";

public static const SLOT_TYPE_SHIELD:String = "BaseShield";

public static const SLOT_TYPE_SPECIAL:String = "Special";

public static const SLOT_TYPE_SPINAL:String = "Spinal";

public static const SLOT_TYPE_TECH:String = "Tech";

public static const SLOT_TYPE_WEAPON:String = "Weapon";

public static const SLOT_TYPE_STRUCTURE:String = "Structure";

public
```

```
static const SLOT_TYPE_TURRET:String = "BaseTurret";

}
}

------------------------------------
File 101: igw\com\enum\StarbaseCategoryEnum.as
package com.enum
{
public class StarbaseCategoryEnum
{
public static const DEFENSE:String = 'Defense';

public static const FLEETS:String = 'Fleets';

public static const INFRASTRUCTURE:String = 'Infrastructure';

public static const RESEARCH:String = 'Research';

public static const STARBASE_STRUCTURE:String = 'StarbaseStructure';
}
}

------------------------------------
File 102: igw\com\enum\StarbaseConstructionEnum.as
package com.enum
{
public class StarbaseConstructionEnum
{
public static const BASE:String = "Base";

public static const DEFENSE:String = "Defense";

public static const PLATFORM:String = "Platform";
}
}

------------------------------------
File 103: igw\com\enum\StatModConditionalEnum.as
package com.enum
{
public class StatModConditionalEnum
{
public static const Unknown:String = "";

public static const eq:String = "==";

public static const ne:String = "!=";

public static const gt:String = ">";

public
```

```
static const lt:String = "<";

public static const ge:String = ">=";

public static const le:String = "<=";
}
}

------------------------------------
File 104: igw\com\enum\StatModScopeEnum.as
package com.enum
{
public class StatModScopeEnum
{
public static const Unknown:uint = 0;

public static const Module:uint = 1;

public static const Entity:uint = 2;

public static const Area:uint = 3;

public static const Player:uint = 4;
}
}
------------------------------------
File 105: igw\com\enum\TimeLogEnum.as
package com.enum
{
public class TimeLogEnum
{
public static const CHARACTER_SELECT:String = "CHARACTER_SELECT";

public static const CONNECT_TO_PROXY:String = "CONNECT_TO_PROXY";

public static const FACTION_SELECT:String = "FACTION_SELECT";

public static const FILE_LOAD:String = "FILE_LOAD";

public static const GAME_INITIALIZED:String = "GAME_INITIALIZED";

public static const GAME_LOAD:String = "GAME_LOAD";

public static const JSON_PARSE:String = "JSON_PARSE";

public static const LOGIN_TO_ACCOUNT:String = "LOGIN_TO_ACCOUNT";

public static const PRELOAD_SCREEN:String = "PRELOAD_SCREEN";

public static const SERVER_GAME_DATA:String = "SERVER_GAME_DATA";

public
```

```
static const SPRITESHEET_PARSE:String = "SPRITESHEET_PARSE";
}
}

------------------------------------
File 106: igw\com\enum\ToastEnum.as
package com.enum
{
import com.event.StateEvent;

public class ToastEnum
{
//types
public static const FLEET_DOCKED:Object = create(2500, 0, 5,
AudioEnum.VO_ALERT_FLEET_DOCKED);
public static const FLEET_REPAIRED:Object = create(2000, 0, 4, null);
public static const FTE_REWARD:Object = create(100000, 0, 3, null,
StateEvent.GAME_BATTLE);
public static const LEVEL_UP:Object = create(5000, 0, 1, AudioEnum.VO_ALERT_LEVEL_UP);
public static const MISSION_NEW:Object = create(2000, 0, 3,
AudioEnum.VO_ALERT_INCOMING_MESSAGE);
public static const MISSION_REWARD:Object = create(3000, 0, 2, null);
public static const TRANSACTION_COMPLETE:Object = create(2000, 0, 4, null,
StateEvent.GAME_STARBASE);
public static const WRONG:Object = create(2800, 1, 5, AudioEnum.AFX_WRONG);
public static const PALLADIUM_ADDED:Object = create(2800, 0, 1,
AudioEnum.AFX_TRANSACTION_PALLADIUM_ADDED);
public static const ALLIANCE:Object = create(2800, 1, 5, null);
public static const BLUEPRINT:Object = create(4000, 0, 5, null);
public static const BUBBLE_ALERT:Object = create(4000, 0, 1, null,
StateEvent.GAME_BATTLE);
public static const BASE_RELOCATED:Object = create(4000, 1, 1, null);
public static const ACHIEVEMENT:Object = create(4000, 1, 1, null);

private static function create( duration:int, limit:int, priority:int, sound:String, state:String = null
):Object
{
var obj:Object = {};
obj.duration = duration;
obj.limit = limit;
obj.priority = priority;
obj.sound = sound;
obj.state = state;
return obj;
}
}
}

------------------------------------
File 107: igw\com\enum\TooltipEnum.as
package
```

```
com.enum
{
public class TooltipEnum
{
public static const AREA:Array = ['itemClass', 'powerCost', 'activated', 'damage', 'outerWidth',
'outerDistance', 'radius', 'targeting', 'resolution', 'penetration', 'tracking', 'attackType',
'damageType', 'chargeTime', 'fireTime', 'reloadTime', 'duration', 'tickRate'];
public static const AREA_ABBR:Array = ['damage', 'outerWidth', 'outerDistance', 'radius',
'damageType'];

public static const ARMOR:Array = ['itemClass', 'powerCost', 'forceArmor', 'explosiveArmor',
'energyArmor'];
public static const ARMOR_ABBR:Array = ['forceArmor', 'explosiveArmor', 'energyArmor'];

public static const BEAM:Array = ['itemClass', 'powerCost', 'activated', 'damage', 'minRange',
'maxRange', 'targeting', 'resolution', 'penetration', 'tracking',
'attackType', 'damageType', 'chargeTime', 'fireTime', 'reloadTime', 'burstSize'];
public static const BEAM_ABBR:Array = ['damage', 'minRange', 'maxRange', 'damageType'];

public static const BUILDING:Array = ['health', 'profile', 'armor', 'masking', 'canBeRecycled'];
public static const BUILDING_ABBR:Array = ['health', 'armor'];

public static const DEFENSE:Array = ['itemClass', 'powerCost', 'guidedIntercept',
'projectileIntercept', 'beamIntercept', 'droneIntercept', 'fireDelay', 'fireRange', 'droneDamage',
'smartTargeting'];
public static const DEFENSE_ABBR:Array = ['guidedIntercept', 'projectileIntercept',
'beamIntercept', 'droneIntercept', 'fireDelay', 'fireRange', 'droneDamage', 'smartTargeting'];

public static const DRONE:Array = ['itemClass', 'powerCost', 'activated', 'damage', 'maxRange',
'targeting', 'resolution', 'penetration', 'tracking', 'attackType', 'damageType', 'launchTime',
'rebuildTime', 'maxDrones', 'damageTime', 'speed', 'lifeTime', 'droneHealth'];
public static const DRONE_ABBR:Array = ['damage', 'maxRange', 'damageType', 'maxDrones'];

public static const PROJECTILE:Array = ['itemClass', 'powerCost', 'activated', 'damage',
'minRange', 'maxRange', 'targeting', 'resolution', 'penetration', 'tracking', 'attackType',
'damageType',
'chargeTime', 'fireTime', 'reloadTime', 'burstSize', 'volleySize', 'speed', 'turnSpeed',
'splashRadius', 'splashDamage','fuseRadius'];
public static const PROJECTILE_ABBR:Array = ['damage', 'minRange', 'maxRange',
'damageType'];

public static const SHIELD:Array = ['itemClass', 'powerCost', 'forceShielding',
'explosiveShielding', 'energyShielding'];
public static const SHIELD_ABBR:Array = ['forceShielding', 'explosiveShielding',
'energyShielding'];

public static const SHIP_RESEARCH:Array = ['health', 'mapSpeed', 'maxSpeed',
'rotationSpeed', 'allowPivot', 'shieldThreshold', 'shieldResetTime', 'profile', 'evasion', 'armor',
'masking', 'loadSpeed', 'cargo', 'power', 'specialSlots', 'weaponSlots', 'defenseSlots', 'techSlots',
'structureSlots'];
public
```

```
static const SHIP_RESEARCH_ABBR:Array = ['health', 'maxSpeed', 'cargo', 'power'];

public static const SHIP_BUILT:Array = ['health', 'shipDps', 'shipRange'];

public static const TECH:Array = ['itemClass', 'powerCost'];
public static const TECH_ABBR:Array = [];
}
}

------------------------------------
File 108: igw\com\enum\TradeRouteQualityEnum.as
package com.enum
{
public class TradeRouteQualityEnum
{
public static const UNKNOWN:int = 0;

public static const COMMON:int = 1;
public static const UNCOMMON:int = 2;
public static const RARE:int = 3;
public static const EPIC:int = 4;
public static const LEGENDARY:int = 5;
}
}
------------------------------------
File 109: igw\com\enum\TypeEnum.as
package com.enum
{
public class TypeEnum
{
//backgrounds
public static const BACKGROUND:String = "Background";
public static const BACKGROUND_ELEMENTS:String = "BackgroundElements";
public static const BACKGROUND_FOG_STARS:String = "BackgroundFogStars";

//defense
public static const ANTI_MISSILE_SYSTEM:String = "AntiMissileSystem";
public static const DEFLECTOR_SCREEN:String = "DeflectorScreen";
public static const DISTORTION_WEB:String = "DistortionWeb";

//projectiles
public static const TURRET_BOMBARDMENT_CANNON:String = "BombardmentCannon";
public static const TURRET_SENTINEL_MOUNT:String = "SentinelMount";
public static const TURRET_MISSILE_POD:String = "MissilePod";
public static const TURRET_BEAM_POINT_DEFENSE_CLUSTER:String = "PDBeam";
public static const TURRET_DRONE:String = "Drone";
public static const IGA_DRONE:String = "IGA_Drone";
public static const SOV_DRONE:String = "SOV_Drone";
public static const TYR_DRONE:String = "Drone";
public static const IGA_MISSILE:String = "IGA_Missile";
public
```

```
static const SOV_MISSILE:String = "SOV_Missile";
public static const TYR_MISSILE:String = "TY_Missile";

//buildings
public static const ADVANCED_TECH:String = "AdvancedTechFacility";
public static const BUILDINGS_DAMAGED:String = "BuildingsDamaged";
public static const COMMAND_CENTER:String = "CommandCenter";
public static const CONDUIT:String = "Conduit";
public static const CONSTRUCTION_BAY:String = "ConstructionBay";
public static const DEFENSE_DESIGN:String = "DefenseDesignFacility";
public static const DOCK:String = "Dock";
public static const OFFICERS_LOUNGE:String = "OfficersLounge";
public static const ACADEMY:String = "Academy";
public static const POINT_DEFENSE_PLATFORM:String = "PointDefensePlatform";
public static const PYLON:String = "Pylon";
public static const REACTOR_STATION:String = "ReactorStation";
public static const RESOURCE_DEPOT:String = "ResourceDepot";
public static const SHIELD_GENERATOR:String = "ShieldGenerator";
public static const SHIPYARD:String = "ShipDesignFacility";
public static const SURVEILLANCE:String = "Surveillance";
public static const WEAPONS_FACILITY:String = "WeaponsDesignFacility";

//sector
public static const ATTACK_ICON:String = "AttackIcon";
public static const DERELICT_IGA:String = "DerelictIGA";
public static const DERELICT_IMPERIUM:String = "DerelictImperium";
public static const DERELICT_SOVEREIGNTY:String = "DerelictSovereignty";
public static const DERELICT_TYRANNAR:String = "DerelictTyrannar";
public static const MISSILE_TRAIL:String = "MissileTrail";
public static const OUTPOST_IGA:String = "OutpostIGA";
public static const OUTPOST_SOVEREIGNTY:String = "OutpostSovereignty";
public static const OUTPOST_TYRANNAR:String = "OutpostTyrannar";
public static const SELECTOR_IGA:String = "SelectorIGA";
public static const SELECTOR_SOVEREIGNTY:String = "SelectorSovereignty";
public static const SELECTOR_TYRANNAR:String = "SelectorTyrannar";
public static const STARBASE_SECTOR_IGA:String = "StarbaseSectorIGA";
public static const STARBASE_SECTOR_SOVEREIGNTY:String =
"StarbaseSectorSovereignty";
public static const STARBASE_SECTOR_TYRANNAR:String = "StarbaseSectorTyrannar";
public static const STARBASE_SHIELD_IGA:String = "StarbaseShieldIGA";
public static const STARBASE_SHIELD_SOVEREIGNTY:String = "StarbaseShieldSovereignty";
public static const STARBASE_SHIELD_TYRANNAR:String = "StarbaseShieldTyrannar";
public static const THRUSTER:String = "Thruster";
public static const MUZZLE:String = "WeaponCharge";
public static const TRADEDEPOT_IGA:String = "TradeDepotIGA";
public static const TRADEDEPOT_SOVEREIGNTY:String = "TradeDepotSovereignty";
public static const TRADEDEPOT_TYRANNAR:String = "TradeDepotTyrannar";
public static const TRANSGATE_IGA:String = "TransgateIGA";
public static const TRANSGATE_SOVEREIGNTY:String = "TransgateSovereignty";
public static const TRANSGATE_TYRANNAR:String = "TransgateTyrannar";

//ships
```

```
public static const FIGHTER:String = "Fighter";
public static const HEAVY_FIGHTER:String = "Heavy Fighter";
public static const CORVETTE:String = "Corvette";
public static const DESTROYER:String = "Destroyer";
public static const BATTLESHIP:String = "Battleship";
public static const DREADNOUGHT:String = "Dreadnought";
public static const TRANSPORT:String = "Transport";

//starbase
public static const STARBASE_IGA:String = "StarbaseIGA";
public static const STARBASE_ARM_IGA:String = "StarbaseArmIGA";
public static const STARBASE_PLATFORMA_IGA:String = "StarbasePlatformAIGA";
public static const STARBASE_PLATFORMB_IGA:String = "StarbasePlatformBIGA";
public static const ISO_1x1_IGA:String = "Iso1x1IGA";
public static const ISO_2x2_IGA:String = "Iso2x2IGA";
public static const ISO_3x3_IGA:String = "Iso3x3IGA";
//
public static const STARBASE_SOVEREIGNTY:String = "StarbaseSovereignty";
public static const STARBASE_ARM_SOVEREIGNTY:String = "StarbaseArmSovereignty";
public static const STARBASE_PLATFORMA_SOVEREIGNTY:String =
"StarbasePlatformASovereignty";
public static const STARBASE_PLATFORMB_SOVEREIGNTY:String =
"StarbasePlatformBSovereignty";
public static const ISO_1x1_SOVEREIGNTY:String = "Iso1x1Sovereignty";
public static const ISO_2x2_SOVEREIGNTY:String = "Iso2x2Sovereignty";
public static const ISO_3x3_SOVEREIGNTY:String = "Iso3x3Sovereignty";
//
public static const STARBASE_TYRANNAR:String = "StarbaseTyrannar";
public static const STARBASE_ARM_TYRANNAR:String = "StarbaseArmTyrannar";
public static const STARBASE_PLATFORMA_TYRANNAR:String =
"StarbasePlatformATyrannar";
public static const STARBASE_PLATFORMB_TYRANNAR:String =
"StarbasePlatformBTyrannar";
public static const ISO_1x1_TYRANNAR:String = "Iso1x1Tyrannar";
public static const ISO_2x2_TYRANNAR:String = "Iso2x2Tyrannar";
public static const ISO_3x3_TYRANNAR:String = "Iso3x3Tyrannar";
//
public static const PYLON_BASE_IGA:String = "PylonBaseIGA";
public static const PYLON_BASE_SOVEREIGNTY:String = "PylonBaseSovereignty";
public static const PYLON_BASE_TYRANNAR:String = "PylonBaseTyrannar";

//
public static const STARBASE_WALL:String = "StarbaseWall";
public static const STARBASE_ARM:String = "StarbaseArm";
public static const STARBASE_PLATFORMA:String = "StarbasePlatformA";
public static const STARBASE_PLATFORMB:String = "StarbasePlatformB";

//visual component
public static const BASE_RANGE:String = "BaseRange";
public
```

```
static const BUILDING_ANIMATION:String = "BuildingAnimation";
public static const BUILDING_CONSTRUCTION:String = "Construction";
public static const BUILDING_SHIELD:String = "BuildingShield";
public static const RESOURCE_DEPOT_CANISTER:String = "ResourceDepotCanister";
public static const NAME:String = "EntityName";
public static const STATE_BAR:String = "StateBar";
public static const HEALTH_BAR:String = "HealthBar";
public static const SHIELD:String = "Shield";
public static const SHIP_RANGE:String = "ShipRange";
public static const SHIP_SELECTION_RANGE:String = "ShipSelectionBox";
public static const STARBASE_TURRET:String = "BaseTurrets";
public static const TURRET_SHIELD:String = "TurretShield";
public static const ATTACK_DEBUFF:String = "General_Attack";
public static const DEFENSE_DEBUFF:String = "General_Defense";
public static const SPEED_DEBUFF:String = "General_Slow";
public static const DEBUFF_TRAY:String = "DebuffTray";

//vfx
public static const DAMAGE:String = "EXPshipdamage";
public static const FORCEFIELD:String = "Forcefield";
public static const EXPLOSION_SMALL:String = "ExplosionSmall";
public static const EXPLOSION_MEDIUM:String = "ExplosionMedium";
public static const EXPLOSION_LARGE:String = "ExplosionLarge";
public static const EXPLOSION_LASERHIT:String = "EXPlaserdamage";
public static const EXPLOSION_FLARE:String = "Flare";
public static const EXPLOSION_SHOCKWAVE:String = "ShockWave";
public static const EXPLOSION_MISSILEHIT:String = "Explosion_MissileHit";
public static const HIT:String = "Hit";
public static const SHIELD_HIT:String = "ShieldHit";

//vfx messages
public static const ROUTE_LINE:String = "RouteLine";

//for tooltip filtering
public static const SHIP_RESEARCH_TT:uint = 1;
public static const PROJECTILE_TT:uint = 2;
public static const PROJECTILE_TURRET_TT:uint = 3;
public static const BEAM_TT:uint = 4;
public static const BEAM_TURRET_TT:uint = 5;
public static const AREA_TT:uint = 6;
public static const DRONE_TT:uint = 7;
public static const DRONE_TURRET_TT:uint = 8;
public static const DEFENSE_TT:uint = 9;
public static const SHIELD_TT:uint = 10;
public static const ARMOR_TT:uint = 11;
public static const TECH_TT:uint = 12;
public static const BUILDING_TT:uint = 13;
public static const BASE_SHIELD_TT:uint = 14;
public static const BASE_TURRET_TT:uint = 15;
public static const SHIP_BUILT_TT:uint = 16;

public
```

```
static const DEBUG_LINE:String = "DebugLine";

//Mission waypoint types
public static const WAYPOINT_TYPE_COLLECT:String = "collect";
public static const WAYPOINT_TYPE_ESCORT:String = "escort";
public static const WAYPOINT_TYPE_SCAN:String = "scan";
}
}
```

File 110: igw\com\enum\server\AllianceRankEnum.as

```
package com.enum.server
{
public class AllianceRankEnum
{
public static var UNAFFILIATED:Number = 0;
public static var RECRUIT:Number = 50;
public static var MEMBER:Number = 100;
public static var OFFICER:Number = 200;
public static var LEADER:Number = 300;
}
}
```

File 111: igw\com\enum\server\AllianceResponseEnum.as

```
package com.enum.server
{
public class AllianceResponseEnum
{
public static var ALLIANCE_CREATED:Number = 1;
public static var ALLIANCE_CREATION_FAILED_NAMEINUSE:Number = 2;
public static var ALLIANCE_CREATION_FAILED_BADNAME:Number = 3;
public static var ALLIANCE_CREATION_FAILED_UNKNOWN:Number = 4;
public static var SET_SUCCESS:Number = 5;
public static var SET_FAILED_TOOLONG:Number = 6;
public static var SET_FAILED_LACKINGRANK:Number = 7;
public static var SET_FAILED_UNKNOWN:Number = 8;
public static var INVITE_FAILED_IGNORED:Number = 9;
public static var INVITE_FAILED_OFFLINE:Number = 10;
public static var INVITE_FAILED_INALLIANCE:Number = 11
public static var JOIN_FAILED_TOOMANYPLAYERS:Number = 12;
public static var JOIN_FAILED_NOALLIANCE:Number = 13;
public static var KICKED:Number = 14;
public static var LEFT:Number = 15;
public static var JOINED:Number = 16
public static var YOUGOTDEMOTED:Number = 17;
public static var INVITED:Number = 100;
}
}
```

File 112: igw\com\enum\server\BattleEntityTypeEnum.as

```
package com.enum.server
{
public class BattleEntityTypeEnum
{
public static const UNKNOWN:int = 0;

public static const SHIP:int = 1;

public static const BUILDING:int = 2;

public static const PLATFORM:int = 3;

public static const PYLON:int = 4;

public static const FORCEFIELD:int = 5;
}
}
```

File 113: igw\com\enum\server\ChatChannelEnum.as

```
package com.enum.server
{
public class ChatChannelEnum
{
public static const UNKNOWN:int = 0;

public static const SYSTEM:int = 1;

public static const WHISPER:int = 2;

public static const SECTOR:int = 3;

public static const ALLIANCE:int = 4;

public static const GROUP:int = 5;

public static const BROADCAST:int = 6;

public static const EVENT:int = 7;

public static const GLOBAL:int = 8;

public static const FACTION:int = 9;

public static const MEMBERS:int = 10;
}
}
```

File 114: igw\com\enum\server\ChatResponseCodeEnum.as
```
package com.enum.server
{
public class ChatResponseCodeEnum
{
public static const UNKNOWN:int = 0;

public static const OK:int = 1;

public static const ROOM_JOINED:int = 2;

public static const ROOM_LEFT:int = 3;

public static const NO_SUCH_PLAYER:int = 4;

public static const NO_SUCH_CHAT_ROOM:int = 5;

public static const YOU_ARE_MUTED:int = 6;

public static const SPAM_DELAY:int = 7;
}
}
```

File 115: igw\com\enum\server\EncodingEnum.as
```
package com.enum.server
{
public class EncodingEnum
{
public static const UNKNOWN:int = 0;
public static const BINARY:int = 1;
public static const JSON:int = 2;
public static const BSON:int = 3;
public static const BINARYZLIBCOMPRESSED:int = 4;
}
}
```

File 116: igw\com\enum\server\OrderEnum.as
```
package com.enum.server
{
public class OrderEnum
{
public static const UNKNOWN:int = 0;

public static const MOVE:int = 1;

public static const ATTACK:int = 2;

public
```

```
static const RECALL:int = 3;

public static const FORCED_RECALL:int = 4;

public static const TRANS_GATE_TRAVEL:int = 5;

public static const SALVAGE:int = 6;

public static const DEFEND:int = 7;

public static const REMOTE_MOVE:int = 8;

public static const TACKLE:int = 9;

public static const FORCE_ATTACK:int = 10;

public static const HALT:int = 11;

public static const WAYPOINT_TRAVEL:int = 12;

public static const WAYPOINT_DOCK:int = 13;
}
}

-------------------------------------
File 117: igw\com\enum\server\ProtocolEnum.as
package com.enum.server
{
public class ProtocolEnum
{
public static const PROXY_CLIENT:int = 0;

public static const SECTOR_CLIENT:int = 1;

public static const BATTLE_CLIENT:int = 2;

public static const STARBASE_CLIENT:int = 3;

public static const MAIL_CLIENT:int = 4;

public static const ALLIANCE_CLIENT:int = 5;

public static const CHAT_CLIENT:int = 6;

public static const LEADERBOARD_CLIENT:int = 7;

public static const UNIVERSE_CLIENT:int = 8;
}
}

-------------------------------------
```

File 118: igw\com\enum\server\PurchaseTypeEnum.as

```
package com.enum.server
{
public class PurchaseTypeEnum
{
public static const NORMAL:int = 0;
public static const INSTANT:int = 1;
public static const GET_RESOURCES:int = 2;
}
}
```

File 119: igw\com\enum\server\ReplicableOpEnum.as

```
package com.enum.server
{
public class ReplicableOpEnum
{
public static var EndDeltas:int = -128;
public static var Copy:int = 127;
public static var Modifychild:int = 100;
public static var Insertdefault:int = 101; // operator[] on a map can cause this
public static var Set:int = 102;
public static var Erase:int = 103;
public static var Clear:int = 104;
public static var Pushback:int = 105; // vector api
}
}
```

File 120: igw\com\enum\server\RequestEnum.as

```
package com.enum.server
{
public class RequestEnum
{
public static const BINARY_OBJECT_SEQUENCE_TOKEN_START:int = 1000000000;
public static const BINARY_OBJECT_SEQUENCE_TOKEN_END:int = 1000000001;


//=========================================================================
//============== PROXY_CLIENT PROTOCOLS
==================================

//=========================================================================
public static const PROXY_TIME_SYNC:int = 1;
public static const PROXY_CONNECT_TO_BATTLE:int = 2;
public static const PROXY_CONNECT_TO_SECTOR:int = 3;
public static const PROXY_LOGIN:int = 4;
public static const PROXY_REPORT_CRASH:int = 5;
public static const PROXY_REPORT_LOGIN_DATA:int = 6;
public static const PROXY_TUTORIAL_STEP_COMPLETED:int = 7;


//=========================================================================
```

```
//=============== SECTOR_CLIENT PROTOCOLS
====================================

//=============================================================================
public static const AUTHORIZATION:int = 1000;

public static const SECTOR_SET_VIEW_LOCATION:int = 1;
public static const SECTOR_ISSUE_ORDER:int = 2;

public static const SECTOR_REQUEST_BASELINE:int = 5;


//=============================================================================
//=============== BATTLE_CLIENT PROTOCOLS
====================================

//=============================================================================
public static const BATTLE_MOVE_ORDER:int = 1;
public static const BATTLE_ATTACK_ORDER:int = 2;
public static const BATTLE_TOGGLE_MODULE_ORDER:int = 3;
public static const BATTLE_PAUSE:int = 4;
public static const BATTLE_RETREAT:int = 5;


//=============================================================================
//=============== STARBASE_CLIENT PROTOCOLS
==================================

//=============================================================================
public static const STARBASE_BUILD_SHIP:int = 1;
public static const STARBASE_UPDATE_FLEET:int = 2;
public static const STARBASE_LAUNCH_FLEET:int = 3;
public static const STARBASE_RECALL_FLEET:int = 4;
public static const STARBASE_REPAIR_FLEET:int = 5;
public static const STARBASE_RENAME_FLEET:int = 6;
public static const STARBASE_BUILD_NEW_BUILDING:int = 7;
//public static const STARBASE_CREATE_CHARACTER:int = 8;
public static const STARBASE_SET_CLIENT_SETTINGS:int = 9;
public static const STARBASE_UPGRADE_BUILDING:int = 10;
public static const STARBASE_RECYCLE_BUILDING:int = 11;
public static const STARBASE_REFIT_BUILDING:int = 12;
public static const STARBASE_REPAIR_BASE:int = 13;
public static const STARBASE_SPEED_UP_TRANSACTION:int = 14;
public static const STARBASE_CANCEL_TRANSACTION:int = 15;
public static const STARBASE_MOVE_BUILDING:int = 16;
public static const STARBASE_RESEARCH:int = 17;
public static const STARBASE_BUY_RESOURCE:int = 18;
public static const STARBASE_BUY_STORE_ITEM:int = 19;
public static const STARBASE_RECYCLE_SHIP:int = 20;
public
```

```
static const STARBASE_REFIT_SHIP:int = 21;
public static const STARBASE_NEGOTIATE_CONTRACT:int = 22;
public static const STARBASE_BRIBE_CONTRACT:int = 23;
public static const STARBASE_CANCEL_CONTRACT:int = 24;
public static const STARBASE_EXTEND_CONTRACT:int = 25;
public static const STARBASE_RESECURE_CONTRACT:int = 26;
public static const STARBASE_MISSION_STEP:int = 27;
public static const STARBASE_MISSION_ACCEPT:int = 28;
public static const STARBASE_MISSION_ACCEPT_REWARDS:int = 29;
public static const STARBASE_BUYOUT_BLUEPRINT:int = 30;
public static const STARBASE_BATTLELOG_LIST:int = 32;
public static const STARBASE_BATTLELOG_DETAILS:int = 33;
public static const STARBASE_BOOKMARK_SAVE:int = 34;
public static const STARBASE_BOOKMARK_DELETE:int = 35;
public static const STARBASE_BOOKMARK_UPDATE:int = 36;
public static const STARBASE_MARK_MOTD_READ_MESSAGE:int = 37;
public static const STARBASE_CLAIM_DAILY_MESSAGE:int = 38;
public static const STARBASE_SKIP_TRAINING_MESSAGE:int = 39;
public static const STARBASE_REROLL_BLUEPRINT_CHANCE_MESSAGE:int = 40;
public static const STARBASE_REROLL_BLUEPRINT_RECEIVED_MESSAGE:int = 41;
public static const STARBASE_RENAME_PLAYER:int = 42;
public static const STARBASE_MOVE_STARBASE:int = 43;
public static const STARBASE_REQUEST_ACHIEVEMENTS:int = 44;
public static const STARBASE_CLAIM_ACHIEVEMENT_REWARD:int = 45;
public static const STARBASE_GET_PAYWALL_PAYOUTS:int = 46;
public static const STARBASE_VERIFY_PAYMENT:int = 47;
public static const STARBASE_BUY_OTHER_STORE_ITEM:int = 49;
public static const STARBASE_INSTANCED_MISSION_START:int = 50;
public static const STARBASE_MOVE_STARBASE_TO_TRANSGATE:int = 51;
public static const STARBASE_COMPLETE_BLUEPRINT_RESEARCH:int = 52;
public static const STARBASE_REQUEST_ALL_SCORES:int = 53;
public static const STARBASE_MINT_NFT:int = 54;



//=============================================================================
//============== MAIL_CLIENT PROTOCOLS
==================================

//=============================================================================
public static const MAIL_REQUEST_INBOX:int = 1;
public static const MAIL_SEND_MAIL:int = 2;
public static const MAIL_DELETE_MAIL:int = 3;
public static const MAIL_READ_MAIL:int = 4;
public static const MAIL_SEND_ALLIANCE_MAIL:int = 5;


//=============================================================================
//============== ALLIANCE_CLIENT PROTOCOLS
==================================

//=============================================================================
```

```
public static const ALLIANCE_REQUEST_BASELINE:int = 1;
public static const ALLIANCE_REQUEST_ROSTER:int = 2;
public static const ALLIANCE_CREATE_ALLIANCE:int = 3;
public static const ALLIANCE_SET_MOTD:int = 4;
public static const ALLIANCE_SET_DESCRIPTION:int = 5;
public static const ALLIANCE_SET_PUBLIC:int = 6;
public static const ALLIANCE_PROMOTE:int = 7;
public static const ALLIANCE_DEMOTE:int = 8;
public static const ALLIANCE_KICK:int = 9;
public static const ALLIANCE_LEAVE_ALLIANCE:int = 10;
public static const ALLIANCE_JOIN_ALLIANCE:int = 11;
public static const ALLIANCE_SEND_INVITE:int = 12;
public static const ALLIANCE_IGNORE_INVITES:int = 13;
public static const ALLIANCE_REQUEST_PUBLICS:int = 14;



//=========================================================================
//============== CHAT_CLIENT PROTOCOLS
==================================

//=========================================================================
public static const CHAT_SEND_CHAT:int = 1;
public static const CHAT_IGNORE_CHAT:int = 2;
public static const CHAT_REPORT_CHAT:int = 3;
public static const CHAT_CHANGE_ROOM:int = 4;


//=========================================================================
//============== LEADERBOARD_CLIENT PROTOCOLS
==================================

//=========================================================================
public static const LEADERBOARD_REQUEST_LEADERBOARD:int = 1;
public static const LEADERBOARD_REQUEST_PLAYER_PROFILE:int = 2;



//=========================================================================
//============== UNIVERSE_CLIENT PROTOCOLS
==================================

//=========================================================================
public static const UNIVERSE_CHARACTER_CREATION_REQUEST:int = 1;
}
}

----------------------------------
File
```

121: igw\com\enum\server\ResponseEnum.as
```
package com.enum.server
{
public class ResponseEnum
{
//Proxy
public static const PROXY_TIME_SYNC:int = 1;
public static const PROXY_BATTLE_DISCONNECTED:int = 2;
public static const PROXY_SECTOR_DISCONNECTED:int = 3;
public static const PROXY_STARBASE_DISCONNECTED:int = 4;

public static const AUTHORIZATION:int = 1000;

//Sector
public static const SECTOR_BASELINE:int = 1;
public static const SECTOR_UPDATE:int = 2;
public static const SECTOR_ALWAYS_VISIBLE_BASELINE:int = 3;
public static const SECTOR_ALWAYS_VISIBLE_UPDATE:int = 4;
public static const SECTOR_FLEET_TRAVEL_ALERT:int = 6;

//Battle
public static const BATTLE_BASELINE:int = 1;
public static const BATTLE_UPDATE:int = 2;
public static const BATTLE_DEBUG_LINES:int = 3;
public static const BATTLE_OBJECT_UPDATE:int = 4;
public static const BATTLE_START_TIME:int = 5;
public static const BATTLE_HAS_BEGUN:int = 6;
public static const BATTLE_HAS_ENDED:int = 7;

//Starbase
public static const STARBASE_BASELINE:int = 1;
public static const STARBASE_TRANSACTION_RESPONSE:int = 2;

public static const STARBASE_BATTLE_ALERT:int = 4;
public static const STARBASE_MISSION_COMPLETE:int = 5;
public static const STARBASE_FLEET_DOCKED:int = 6;
public static const STARBASE_BOUNTY_REWARD:int = 7;
public static const STARBASE_BATTLELOG_LIST:int = 8;
public static const STARBASE_BATTLELOG_DETAILS:int = 9;
public static const STARBASE_OFFER_REDEEMED:int = 10;
public static const STARBASE_MOTD_LIST:int = 11;
public static const STARBASE_DAILY:int = 12;
public static const STARBASE_DAILY_REWARD:int = 13;
public static const STARBASE_AVAILABLE_REROLL:int = 14;
public static const STARBASE_REROLL_CHANCE_RESULT:int = 15;
public static const STARBASE_REROLL_RECEIVED_RESULT:int = 16;
public static const STARBASE_MOVE_STARBASE_RESPONSE:int = 17;
public static const STARBASE_ACHIEVEMENTS_RESPONSE:int = 18;
public static const STARBASE_GET_PAYWALL_PAYOUTS_RESPONSE:int = 19;
public static const STARBASE_UNAVAILABLE_REROLL:int = 20;
public
```

```
static const STARBASE_AVAILABLE_CREWMEMBER_REROLL:int = 21;
public static const STARBASE_UNAVAILABLE_CREWMEMBER_REROLL:int = 22;
public static const STARBASE_REROLL_CREWMEMBER_RECEIVED_RESULT:int = 23;
public static const STARBASE_INSTANCED_MISSION_ALERT:int = 24;
public static const STARBASE_ALL_SCORES_RESPONSE:int = 25;

// mail
public static const MAIL_UNREAD:int = 1;
public static const MAIL_INBOX:int = 2;
public static const MAIL_DETAIL:int = 3;

// alliance
public static const ALLIANCE_BASELINE:int = 1;
public static const ALLIANCE_ROSTER:int = 2;
public static const GENERIC_ALLIANCE_RESPONSE:int = 3;
public static const PUBLIC_ALLIANCES_RESPONSE:int = 4;
public static const ALLIANCE_INVITE:int = 5;

// chat
public static const CHAT_BASELINE:int = 1;
public static const CHAT_RESPONSE:int = 2;
public static const CHAT_EVENT:int = 3;

// leaderboard
public static const LEADERBOARD:int = 1;
public static const PLAYER_PROFILE:int = 2;
public static const WARFRONT_UPDATE:int = 3;

// universe
public static const UNIVERSE_NEED_CHARACTER_CREATE:int = 1;
public static const UNIVERSE_SECTOR_LIST:int = 2;

}
}

-------------------------------------
File 122: igw\com\enum\server\SectorEntityStateEnum.as
package com.enum.server
{
public class SectorEntityStateEnum
{
public static const UNKNOWN:int = 0;

public static const STATIONARY:int = 1;

public static const MOVING:int = 2;

public static const BATTLE:int = 3;

public static const LEAVING:int = 4;

public
```

```
static const LOADING:int = 5;

public static const FORCED_RECALLING:int = 6;

public static const SALVAGING:int = 7;

public static const DEFENDING:int = 8;
}
}

---------------------------------
File 123: igw\com\enum\server\SectorEntityTypeEnum.as
package com.enum.server
{
public class SectorEntityTypeEnum
{
public static const UNKNOWN:int = 0;

public static const BASE:int = 1;

public static const FLEET:int = 2;

public static const TRANS_GATE:int = 3;

public static const DEPOT:int = 4;

public static const DERELICT:int = 5;

public static const MISSION_OBJECTIVE:int = 6;
}
}

---------------------------------
File 124: igw\com\enum\server\SpeakerEnum.as
package com.enum.server
{
public class SpeakerEnum
{
public static const SECTOR_SPEAKER:int = 1;
public static const CLIENT_SPEAKER:int = 2;
}
}

---------------------------------
File 125: igw\com\enum\server\StarbaseBuildStateEnum.as
package com.enum.server
{
public class StarbaseBuildStateEnum
{
public static const UNKNOWN:int = 0;

public
```

```
static const DONE:int = 1;

public static const BUILDING:int = 2;

public static const UPGRADING:int = 3;

public static const REFITTING:int = 4;

public static const REPAIRING:int = 5;

public static const RESEARCHING:int = 6;

public static const TRADING:int = 7;
}
}
```

File 126: igw\com\enum\server\StarbaseTransactionStateEnum.as

```
package com.enum.server
{
public class StarbaseTransactionStateEnum
{
public static const PENDING:int = -1;

public static const UNKNOWN:int = 0;

public static const FAILED:int = 1;

public static const SAVING:int = 2;

public static const SAVED:int = 3;

public static const TIMER_RUNNING:int = 4;

public static const TIMER_DONE:int = 5;

public static const CANCELLED:int = 6;
}
}
```

File 127: igw\com\enum\server\UnavailableRerollEnum.as

```
package com.enum.server
{
public class UnavailableRerollEnum
{
public static const ROLL_ALLOWED:int = 1000;

public static const UNKNOWN:int = 0;

public
```

```
static const NO_CHALLENGE:int = 1;

public static const FARM_PENALTY:int = 2;

public static const DAMAGE_REQ:int = 3;

public static const ALL_COMPLETE:int = 4;

public static const NONE_IN_BAND:int = 5;

public static const SLOTS_FULL:int = 6;

public static const MISSING_TECH:int = 7;

public static const BAD_ENCOUNTER:int = 8;

public static const BAD_FACTION:int = 9;

public static const NONE_AT_RARITY:int = 10;
}
}

---
File 128: igw\com\enum\ui\ButtonEnum.as
package com.enum.ui
{
public class ButtonEnum
{
public static const ACCORDIAN_SUBITEM:String = "BtnAccordian";

public static const BLUE_A:String = "BtnBlueA";

public static const BUFF_COMMISSION:String = "BtnBuffPrivateersCommision";

public static const BUFF_MAP_SPEED:String = "BtnBuffMapSpeed";

public static const BUFF_RECLAMATION:String = "BtnBuffReclamationLicense";

public static const BUFF_REPAIR_SPEED:String = "BtnBuffRepairSpeed";

public static const BUFF_RESOURCE:String = "BtnBuffResource";

public static const BUFF_SHIELD:String = "BtnBuffShield";

public static const CHARACTER_FRAME:String = "CharacterFrame";

public static const ICON_FRAME:String = "BtnIconFrame";

public static const CHECKBOX:String = "BtnCheckBox";

public
```

```
static const CHECKBOX_LARGE:String = "BtnLargeCheckBox";

public static const OFFER_CHEST:String = "BeginnersChest";

public static const CLOSE:String = "BtnClose";

public static const DROP_TAB:String = "BtnDropTab";

public static const GREY:String = "BtnGrey";

public static const GOLD_A:String = "BtnGoldA";

public static const GREEN_A:String = "BtnGreen";

public static const HEADER:String = "HeaderRect";

public static const HEADER_NOTCHED:String = "HeaderNotched";

public static const ICON_ALLIANCE:String = "BtnIconAlliance";

public static const ICON_BATTLE_LOG:String = "BtnIconBattleLog";

public static const ICON_BOOKMARKS:String = "BtnIconBookmarks";

public static const ICON_CHAT_CHANNEL:String = "BtnIconChatChannel";

public static const ICON_FIND:String = "BtnIconFind";

public static const ICON_LEADERBOARD:String = "BtnIconLeaderboard";

public static const ICON_MAIL:String = "BtnIconMail";

public static const ICON_MAXIMIZE:String = "BtnIconMaximize";

public static const ICON_MINIMIZE:String = "BtnIconMinimize";

public static const ICON_MISSION:String = "BtnIconMission";

public static const ICON_SETTINGS:String = "BtnIconSettings";

public static const ICON_WARFRONT:String = "BtnIconWarfront";

public static const ICON_FULLSCREEN:String = "BtnIconFullscreen";

public static const PLUS:String = "IconPlus";

public static const RED_A:String = "BtnRedA";

public static const ICON_WINDOW_FULL:String = "BtnIconWindowFull";

public
```

```
static const ICON_WINDOW_HALF:String = "BtnIconWindowHalf";

public static const ICON_WINDOW_MIN:String = "BtnIconWindowMin";

public static const ICON_IGNORE:String = 'BtnIgnore';

public static const FRAME_BLUE:String = "BtnEmpty";

public static const FRAME_GREEN:String = "BtnBuild";

public static const FRAME_GREY:String = "BtnGrey";

public static const FRAME_RED:String = "BtnRepair";

public static const FRAME_GOLD:String = "BtnGold";

public static const SLIDER:String = "BtnSlider";

public static const BACK_ARROW:String = 'BtnBackArrow';

public static const FORWARD_ARROW:String = 'BtnForwardArrow';

public static const ICON_CHANCE:String = 'BtnIconChance';

public static const FLEET:String = 'BtnFleet';

public static const QUEUE:String = 'BtnQueue';

public static const EDIT:String = "BtnEdit";

public static const FLEET_GOTO:String = "BtnGotoFleet";

public static const REPLY_ALL:String = 'BtnAllianceMailReply';

public static const FAQ_UP_ARROW:String = 'BtnFAQUpArrow';

public static const FAQ_DOWN_ARROW:String = 'BtnFAQDownArrow';

public static const STATE_NORMAL:String = "normal";

public static const STATE_OVER:String = "over";

public static const STATE_DOWN:String = "down";

public static const STATE_SELECTED:String = "selected";

public static const STATE_DISABLED:String = "disabled"
}
}

-----------------------------------
```

File 129: igw\com\enum\ui\FontEnum.as

```
package com.enum.ui
{
public class FontEnum
{

}
}
```

File 130: igw\com\enum\ui\LabelEnum.as

```
package com.enum.ui
{
public class LabelEnum
{
public static const H1:String = "H1";

public static const H2:String = "H2";

public static const H3:String = "H3";

public static const H4:String = "H4";

public static const H5:String = "H5";

public static const SUBTITLE:String = "Subtitle";

public static const TITLE:String = "Title";

public static const DEFAULT:String = "Default";

public static const DEFAULT_OPEN_SANS:String = "DefaultOpenSans";

public static const DESCRIPTION:String = "Description";
}
}
```

File 131: igw\com\enum\ui\PanelEnum.as

```
package com.enum.ui
{
public class PanelEnum
{
public static const CONTAINER_BLACK:String = "ContainerBlackBMD";

public static const CONTAINER_DOUBLE_NOTCHED_ARROWS:String = "ContainerBMD";

public static const CONTAINER_RIGHT_NOTCHED_ARROW:String = 'ContainerRightBMD';

public static const CONTAINER_INNER:String = "ContainerInnerBMD";

public
```

```
static const CONTAINER_INNER_DARK:String = "ContainerInnerDarkBMD";

public static const CONTAINER_NOTCHED:String = "ContainerNotchedBMD";

public static const CONTAINER_DOUBLE_NOTCHED:String =
"ContainerDoubleNotchedBMD";

public static const CONTAINER_NOTCHED_RIGHT_SMALL:String =
"ContainerInnerNotchedRightBMD";

public static const CONTAINER_NOTCHED_LEFT_SMALL:String =
'ContainerInnerNotchedLeftBMD';

public static const PLAYER_CONTAINER_NOTCHED:String = 'PlayerWindowBMD';

public static const ENEMY_CONTAINER_NOTCHED:String = 'EnemyWindowBMD';

public static const HEADER:String = "HeaderRectUpBMD";

public static const HEADER_NOTCHED:String = "HeaderNotchedUpBMD";

public static const HEADER_NOTCHED_RIGHT:String = "HeaderNotchedUpBMD";

public static const STATBAR:String = "StatBarBMD";

public static const STATBAR_CONTAINER:String = "StatBarContainerBMD";

public static const STATBAR_GREY:String = "StatBarGreyBMD";

public static const WINDOW:String = "WindowBMD";

public static const WINDOW_HEADER:String = "WindowHeaderBMD";

public static const WINDOW_SIDE_HEADER:String = "WindowSideHeaderBMD";

public static const WINDOW_X_HEADER:String = "WindowCloseHeaderBMD";

public static const CHARACTER_FRAME:String = "CharacterFrameUpBMD";

public static const BLUE_FRAME:String = "BtnFrameEmptyUpBMD";

public static const PLAYER_HEALTH_BG:String = "PlayerHealthBarBGBMD";

public static const INPUT_BOX_BLUE:String = "InputBoxBMD";

public static const INPUT_BOX_GOLD:String = "InputBoxGoldBMD";

public static const BOX_GOLD_GLOW:String = 'IconGoldGlowBMD';

public static const SCROLL_BAR:String = "ScrollBarBMD";

public
```

```
static const NUMBER_BOX:String = "NumberBoxBMD";

public static const GRID_END:String = "LeaderboardGridEndBMD";

public static const LEADERBOARD_ROW:String = "LeaderboardRowBMD";

public static const LEADERBOARD_ROW_GLOW:String = "LeaderboardRowGlowBMD";

public static const FAQ_SUBJECT_BG:String = 'FAQSubjectBGBMD';

}
}

------------------------------------
File 132: igw\com\event\BattleEvent.as
package com.event
{
import com.service.server.IResponse;

import flash.events.Event;

public class BattleEvent extends Event
{
public static const BATTLE_COUNTDOWN:String = "countdown";
public static const BATTLE_JOIN:String = "join";
public static const BATTLE_STARTED:String = "started";
public static const BATTLE_ENDED:String = "ended";
public static const BATTLE_REPLAY:String = "replay";


public var baseID:String;
public var battleServerAddress:String;
public var response:IResponse;

public function BattleEvent( type:String, battleServerAddress:String = null, response:IResponse
= null )
{
super(type, false, false);
this.battleServerAddress = battleServerAddress;
this.response = response;
}
}
}

------------------------------------
File 133: igw\com\event\FTEEvent.as
package com.event
{
import com.controller.fte.FTEStepVO;

import
```

```
flash.events.Event;

public class FTEEvent extends Event
{
public static const FTE_COMPLETE:String = "FTEComplete";
public static const FTE_STEP:String = "FTEStep";

public var step:FTEStepVO;

public function FTEEvent( type:String, step:FTEStepVO )
{
super(type, false, false);
this.step = step;
}
}
}
```

File 134: igw\com\event\LoadEvent.as

```
package com.event
{

import com.service.loading.loaditems.ILoadItem;

import flash.events.Event;

public final class LoadEvent extends Event
{

//////////////////////////////////////////////////////////
// CONSTANTS
//////////////////////////////////////////////////////////

public static const LOCALIZATION_COMPLETE:String = "LocalizationComplete";

public static const COMPLETE:String = "complete";

public var loadItem:ILoadItem;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function LoadEvent( loadItem:ILoadItem )
{
super(COMPLETE);
this.loadItem = loadItem;
}

//////////////////////////////////////////////////////////
//
```

```
PUBLIC API
//////////////////////////////////////////////////////////

public override function clone():Event
{
return new LoadEvent(loadItem);
}
}
}

-------------------------------------
File 135: igw\com\event\MissionEvent.as
package com.event
{
import flash.events.Event;

public class MissionEvent extends Event
{
public static const MISSION_FAILED:String = "MissionFailed";
public static const MISSION_GREETING:String = "MissionGreeting";
public static const MISSION_SITUATIONAL:String = "MissionSituational";
public static const MISSION_VICTORY:String = "MissionVictory";
public static const SHOW_REWARDS:String = "ShowRewards";

public function MissionEvent( type:String )
{
super(type, false, false);
}
}
}
-------------------------------------
File 136: igw\com\event\PaywallEvent.as
package com.event
{
import flash.events.Event;

public class PaywallEvent extends Event
{
public static const GET_PAYWALL:String = "PayWallGet";
public static const OPEN_PAYWALL:String = "PayWallOpen";
public static const BUY_ITEM:String = 'PayWallBuyItem'

//Paywall Open
public var paywallData:String;

//Paywall Purchase
public var externalTrkid:String;
public var payoutId:String;
public var responseData:String;
public var responseSignature:String;

public
```

```
function PaywallEvent( type:String )
{
super(type, false, false);
}
}
}
```

File 137: igw\com\event\RequestLoadEvent.as

```
package com.event
{
import flash.events.Event;

public class RequestLoadEvent extends Event
{
public static const REQUEST:String = "requestLoad";

public var absoluteURL:Boolean = false;
public var url:String;
public var urls:Array;
public var batchType:int;
public var priority:int;

public function RequestLoadEvent( url:String = null, priority:int = 3, absolute:Boolean = false )
{
super(REQUEST, false, false);
absoluteURL = absolute;
this.url = url;
this.priority = priority;
}

public function batchLoad( batchType:int, urls:Array ):void
{
this.urls = urls;
this.batchType = batchType;
}
}
}
```

File 138: igw\com\event\SectorEvent.as

```
package com.event
{
import flash.events.Event;

public class SectorEvent extends Event
{
public static const CHANGE_SECTOR:String = "ChangeSector";

public var sector:String;
public
```

```
var focusFleetID:String;
public var focusX:int;
public var focusY:int;

public function SectorEvent( type:String, sector:String = null, focusFleetID:String = null,
focusX:int = 0, focusY:int = 0 )
{
super(type, false, false);
this.sector = sector;
this.focusFleetID = focusFleetID;
this.focusX = focusX;
this.focusY = focusY;
}
}
}
```

File 139: igw\com\event\ServerEvent.as

```
package com.event
{
import com.service.server.IResponse;

import flash.events.Event;

public class ServerEvent extends Event
{
public static const NOT_CONNECTED:String = "NotConnected";

public static const CONNECT_TO_PROXY:String = "ConnectToProxy";

public static const LOGIN_TO_ACCOUNT:String = "LoginToAccount";

public static const AUTHORIZED:String = "Authorized";

public static const NEED_CHARACTER_CREATE:String = "NeedCharacterCreate";

public static const FAILED_TO_CONNECT:String = "FailedToConnect";

public static const MAINTENANCE:String = "Maintenance";

public static const SUSPENSION:String = "Suspension";

public static const BANNED:String = "Banned";

public static const OPEN_PAYMENT:String = "OpenPayment";

public static const GUEST_RESTRICTION:String = "GuestRestriction";

private
```

```
var _response:IResponse;

public function ServerEvent( type:String, response:IResponse = null )
{
super(type, false, false);
_response = response;
}

public function get response():IResponse { return _response; }
}
}

-------------------------------------
File 140: igw\com\event\StarbaseEvent.as
package com.event
{
import flash.events.Event;

public class StarbaseEvent extends Event
{
public static const ALERT_FLEET_BATTLE:String = "alertFleetBattle";
public static const ALERT_STARBASE_BATTLE:String = "alertStarbaseBattle";
public static const ALERT_INSTANCED_MISSION_BATTLE:String =
"alertInstancedMissionBattle";

public static const ENTER_BASE:String = "EnterBase";
public static const ENTER_INSTANCED_MISSION:String = "EnterInstancedMission";
public static const WELCOME_BACK:String = "WelcomeBack";

public var baseID:String;
public var battleServerAddress:String;
public var fleetID:String;
public var view:Class;
public var viewData:*;

public function StarbaseEvent( type:String, baseID:String = null )
{
super(type, false, false);
this.baseID = baseID;
}
}
}

-------------------------------------
File 141: igw\com\event\StateEvent.as
package com.event
{
import flash.events.Event;

public class StateEvent extends Event
{
```

```
public static const STARTUP_COMPLETE:String = 'startupComplete';

public static const PRELOAD:String = 'preload';

public static const PRELOAD_COMPLETE:String = 'preloadComplete';

public static const CREATE_CHARACTER:String = 'createCharacter';

public static const GAME_STARBASE:String = 'gameStarbase';

public static const GAME_STARBASE_CLEANUP:String = 'gameStarbaseCleanup';

public static const GAME_BATTLE_INIT:String = 'gameBattleInit';

public static const GAME_BATTLE:String = 'gameBattle';

public static const GAME_BATTLE_CLEANUP:String = 'gameBattleCleanup';

public static const GAME_SECTOR_INIT:String = 'gameSectorInit';

public static const GAME_SECTOR:String = 'gameSector';

public static const GAME_SECTOR_CLEANUP:String = 'gameSectorCleanup';

public static const GAME_SHUTDOWN:String = 'gameShutdown';

public static const DEFAULT_CLEANUP:String = 'defaultCleanup';

public static const LOST_CONTEXT:String = 'gameLostContext';

public static const SHUTDOWN_START:String = "shutdownStart";

public static const SHUTDOWN_FINISH:String = "shutdownFinish";

public var nextState:String;

public function StateEvent( type:String, nextState:String = null )
{
super(type, false, false);
this.nextState = nextState;
}


}
}
```

File 142: igw\com\event\ToastEvent.as

```
package com.event
{
```

```
import com.model.prototype.IPrototype;
import com.model.transaction.TransactionVO;

import flash.events.Event;

public class ToastEvent extends Event
{
public static const SHOW_TOAST:String = "showToast";

public var data:*;
public var prototype:IPrototype;
public var strings:Vector.<String> = new Vector.<String>;
public var transaction:TransactionVO;
public var toastType:Object;

public function ToastEvent()
{
super(SHOW_TOAST, false, false);
}

public function addStrings( ... text ):void
{
for (var i:int = 0; i < text.length; i++)
{
strings.push(text[i]);
}
}

public function addStringsFromArray( text:Array ):void
{
for (var i:int = 0; i < text.length; i++)
{
strings.push(text[i]);
}
}

public function get nextString():String
{
if (strings.length > 0)
return strings.shift();
return '';
}

public function destroy():void
{
prototype = null;
strings.length = 0;
toastType = null;
}
}
```

```
}

-----------------------------------
File 143: igw\com\event\TransactionEvent.as
package com.event
{
import com.model.transaction.TransactionVO;

import flash.events.Event;

public class TransactionEvent extends Event
{
public static const STARBASE_BUILDING_BUILD:String = "BuildingBuild";
public static const STARBASE_BUILDING_MOVE:String = "BuildingMove";
public static const STARBASE_BUILDING_RECYCLE:String = "BuildingRecycle";
public static const STARBASE_BUILDING_UPGRADE:String = "BuildingUpgrade";
public static const STARBASE_BUILD_SHIP:String = "BuildShip";
public static const STARBASE_LAUNCH_FLEET:String = "LaunchFleet";
public static const STARBASE_RENAME_FLEET:String = "RenameFleet";
public static const STARBASE_REPAIR_FLEET:String = "RepairFleet";
public static const STARBASE_RECALL_FLEET:String = "RecallFleet";
public static const STARBASE_REPAIR_BASE:String = "RepairBase";
public static const STARBASE_REFIT_BUILDING:String = "RefitBuilding";
public static const STARBASE_UPDATE_FLEET:String = "UpdateFleet";
public static const STARBASE_SPEED_UP_TRANSACTION:String = "SpeedUpTransaction";
public static const STARBASE_CANCEL_TRANSACTION:String = "CancelTransaction";
public static const STARBASE_BUY_RESOURCES:String = "BuyResources";
public static const STARBASE_BUY_STORE_ITEM:String = "BuyStoreItem";
public static const STARBASE_BUY_OTHER_STORE_ITEM:String = "BuyOtherStoreItem";
public static const STARBASE_RESEARCH:String = "Research";
public static const STARBASE_RECYCLE_SHIP:String = "RecycleShip";
public static const STARBASE_REFIT_SHIP:String = "RefitShip";
public static const STARBASE_NEGOTIATE_CONTRACT_REQUEST:String = "SNCR";
public static const STARBASE_CANCEL_CONTRACT_REQUEST:String = "SCCR";
public static const STARBASE_MISSION_STEP:String = "SBMS"
public static const STARBASE_MISSION_ACCEPT:String = "SBMA";
public static const STARBASE_MISSION_ACCEPT_REWARD:String = "SBMAR";
public static const STARBASE_BLUEPRINT_PURCHASE:String = "SBBPP";
public static const STARBASE_REROLL_BLUEPRINT_CHANCE:String = "SBRBC"
public static const STARBASE_REROLL_RECIEVED_BLUEPRINT:String = "SBRRB";
public static const STARBASE_RENAME_PLAYER:String = "SBRP";
public static const STARBASE_RELOCATE_STARBASE:String = "SBRSB";
public static const STARBASE_INSTANCED_MISSION_START:String = "SIMS";
public static const STARBASE_MINT_NFT:String = "SMNFT";


public var clientData:Object;
public var responseData:TransactionVO;
public var transactionToken:int;

public
```

```
function TransactionEvent( type:String, transactionToken:int, clientData:Object,
responseData:TransactionVO )
{
super(type, false, false);
this.clientData = clientData;
this.responseData = responseData;
this.transactionToken = transactionToken;
}
}
}
```

File 144: igw\com\event\TransitionEvent.as

```
package com.event
{
import flash.events.Event;

public class TransitionEvent extends Event
{
public static const TRANSITION_BEGIN:String = 'transitionBegin';

public static const TRANSITION_FAILED:String = 'transitionFailed';

public static const TRANSITION_COMPLETE:String = 'transitionComplete';

public var initEvent:StateEvent;
public var cleanupEvent:StateEvent;

public function TransitionEvent( type:String )
{
super(type, false, false);
}

public function addEvents( initEvent:StateEvent, cleanupEvent:StateEvent ):void
{
this.initEvent = initEvent;
this.cleanupEvent = cleanupEvent;
}
}
}
```

File 145: igw\com\event\signal\InteractSignal.as

```
package com.event.signal
{
import org.osflash.signals.Signal;

public class InteractSignal extends Signal
{
public static const CLICK:String = "click";
public
```

```
static const RESOLUTION_CHANGE:String = "resolutionChange";
public static const SCROLL:String = "scroll";
public static const ZOOM:String = "zoom";

public function InteractSignal()
{
super(String, Number, Number);
}

public function click( dx:Number, dy:Number ):void
{
dispatch(CLICK, dx, dy);
}

public function resolutionChange():void
{
dispatch(RESOLUTION_CHANGE, 0, 0);
}

public function scroll( dx:Number, dy:Number ):void
{
dispatch(SCROLL, dx, dy);
}

public function zoom( scale:Number ):void
{
dispatch(ZOOM, scale, 0);
}
}
}

-------------------------------------
File 146: igw\com\event\signal\QuadrantSignal.as
package com.event.signal
{
import com.util.rtree.RRectangle;

import flash.geom.Point;
import flash.geom.Rectangle;

import org.osflash.signals.Signal;

public class QuadrantSignal extends Signal
{
public static const VISIBLE_HASH_CHANGED:int = 0;

public function QuadrantSignal()
{
super(int, Rectangle);
}

public
```

```
function visibleHashChanged( viewBounds:Rectangle ):void
{
dispatch(VISIBLE_HASH_CHANGED, viewBounds);
}
}
}
```

File 147: igw\com\event\signal\TransactionSignal.as

```
package com.event.signal
{
import com.model.transaction.TransactionVO;

import org.osflash.signals.Signal;

public class TransactionSignal
{
public static const ALL:int = 0;
public static const DATA_IMPORTED:int = 1;
public static const TRANSACTION:int = 2;
public static const TRANSACTION_REMOVED:int = 3;
public static const TRANSACTION_UPDATED:int = 4;

private var _onDataImported:Signal;
private var _onTransactionRemoved:Signal;
private var _onTransactionUpdated:Signal;
private var _transaction:TransactionVO;

public function TransactionSignal()
{
_onDataImported = new Signal(TransactionVO);
_onTransactionRemoved = new Signal(TransactionVO);
_onTransactionUpdated = new Signal(TransactionVO);
}

public function add( type:int, listener:Function ):void
{
switch (type)
{
//Be notified when new data is imported or a transaction is removed or updated
case ALL:
_onDataImported.add(listener);
_onTransactionRemoved.add(listener);
_onTransactionUpdated.add(listener);
break;

//Be notified when new data is imported from the server
case DATA_IMPORTED:
_onDataImported.add(listener);
break;

//Be
```

```
notified when a transaction is removed
case TRANSACTION_REMOVED:
_onTransactionRemoved.add(listener);
break;

//Be notified when a transaction is updated
case TRANSACTION_UPDATED:
_onTransactionUpdated.add(listener);
break;

//Be notified when a transaction is removed or updated
case TRANSACTION:
_onTransactionRemoved.add(listener);
_onTransactionUpdated.add(listener);
break;
}
}

public function dispatch( type:int, transaction:TransactionVO ):void
{
switch (type)
{
//Dispatch all notifications with the last transaction that was completed
case ALL:
_onDataImported.dispatch(transaction);
_onTransactionRemoved.dispatch(transaction);
_onTransactionUpdated.dispatch(transaction);
break;

//Dispatch a notification that new data has been imported
case DATA_IMPORTED:
_onDataImported.dispatch(transaction);
break;

//Dispatch a notification that a transaction has been removed
case TRANSACTION_REMOVED:
_onTransactionRemoved.dispatch(transaction);
break;

//Dispatch a notification that a transaction has been updated
case TRANSACTION_UPDATED:
_onTransactionUpdated.dispatch(transaction);
break;

//Dispatch a notification that a transaction has been updated and removed
case TRANSACTION:
_onTransactionRemoved.dispatch(transaction);
_onTransactionUpdated.dispatch(transaction);
break;
}
}
```

```
/**
* Removes a listener from all signals
*/
public function remove( listener:Function ):void
{
_onDataImported.remove(listener);
_onTransactionRemoved.remove(listener);
_onTransactionUpdated.remove(listener);
}
}
}


-----------------------------------
File 148: igw\com\game\entity\components\battle\ActiveDefense.as
package com.game.entity.components.battle
{
import org.ash.core.Entity;

public class ActiveDefense
{
public static const BEAM:int = 0;
public static const FLAK:int = 1;
public static const SHIELD:int = 2;

public var alphaDelta:Number;
public var alphaStart:Number;
public var animationLength:Number;
public var animationTime:Number;
public var baseWidth:int = 256;
public var followShipRotation:Boolean;
public var growSpeed:Number;
public var hitLocationX:int;
public var hitLocationY:int;
public var owner:Entity;
public var ownerID:String;
public var ready:Boolean;
public var scaleDelta:Number;
public var scaleStart:Number;
public var sourceAttachPoint:String;
public var strength:Number;
public var type:int;

public function init( type:int, owner:Entity, sourceAttachPoint:String ):void
{
this.animationLength = animationLength;
alphaDelta = alphaStart = animationTime = 0;
this.followShipRotation = false;
this.owner
```

```
= owner;
this.ownerID = owner.id;
this.growSpeed = 1;
this.strength = 0;
this.sourceAttachPoint = sourceAttachPoint;
hitLocationX = hitLocationY = 0;
ready = false;
this.type = type;

if (type == BEAM)
{
animationLength = .4;
alphaStart = 1;
alphaDelta = -1;
} else if (type == FLAK)
{
animationLength = .25;
alphaStart = 1;
alphaDelta = 1;
scaleDelta = .5;
scaleStart = .5;
}
}

public function destroy():void
{
owner = null;
}
}
}

-----------------------------------
File 149: igw\com\game\entity\components\battle\Area.as
package com.game.entity.components.battle
{
import com.model.prototype.IPrototype;

public class Area
{
public var duration:Number;
public var ownerID:String;
public var strength:Number;
public var sourceAttachPoint:String;
public var maxRange:Number;
public var moveWithSource:Boolean;
public var rotateWithSource:Boolean;
public var useBeamDynamics:Boolean;
public var startScaleX:Number;
public var startScaleY:Number;
public var startAlpha:Number;
public
```

```
var animLength:Number;
public var animTime:Number;

private var _alphaDelta:Number;
private var _prototype:IPrototype;
private var _scaleDeltaX:Number;
private var _scaleDeltaY:Number;

public function init( ownerID:String, sourceAttachPt:String, maxRange:Number,
prototype:IPrototype ):void
{
this.ownerID = ownerID;
this.strength = 0;
this.sourceAttachPoint = sourceAttachPt;
this.maxRange = maxRange;
this.animTime = 0;
_prototype = prototype;
resetAnimation();
}

public function resetAnimation():void
{
animTime = 0;
strength = 0;
moveWithSource = _prototype.getValue("moveWithSource");
rotateWithSource = _prototype.getValue("rotateWithSource");
useBeamDynamics = _prototype.getValue("useBeamDynamics");
startScaleX = _prototype.getValue("startScaleX");
endScaleX = _prototype.getValue("endScaleX");
startScaleY = _prototype.getValue("startScaleY");
startAlpha = _prototype.getValue("startAlpha");
endAlpha = _prototype.getValue("endAlpha");
endScaleY = _prototype.getValue("endScaleY");
animLength = _prototype.getValue("animLength");
duration = _prototype.getUnsafeValue("duration") == null ? 0 :
_prototype.getUnsafeValue("duration");
}

public function set endScaleX( v:Number ):void { _scaleDeltaX = v - startScaleX; }
public function set endScaleY( v:Number ):void { _scaleDeltaY = v - startScaleY; }
public function set endAlpha( v:Number ):void { _alphaDelta = v - startAlpha; }

public function get scaleDeltaX():Number { return _scaleDeltaX; }
public function get scaleDeltaY():Number { return _scaleDeltaY; }
public function get alphaDelta():Number { return _alphaDelta; }

public function destroy():void
{
_prototype = null;
}
}
```

```
}

------------------------------------
File 150: igw\com\game\entity\components\battle\Attack.as
package com.game.entity.components.battle
{
import com.service.server.incoming.data.SectorBattleData;

public class Attack
{
public var attackData:*;
public var battleServerAddress:String;
public var bubbled:Boolean = false;
public var inBattle:Boolean = false;

private var _organicTargetID:String;
private var _targetID:String;

public var battle:SectorBattleData;

public function get organicTargetID():String { return _organicTargetID; }
public function set organicTargetID( v:String ):void { _organicTargetID = v; }

public function get targetID():String { return _targetID; }
public function set targetID( v:String ):void { _targetID = v; }

public function destroy():void
{
attackData = null;
battleServerAddress = null;
bubbled = false;
inBattle = false;
_organicTargetID = null;
_targetID = null;
battle = null;
}
}
}

------------------------------------
File 151: igw\com\game\entity\components\battle\Beam.as
package com.game.entity.components.battle
{
public class Beam
{
public var attackHit:Boolean;
public var baseWidth:int = 256;
public var followShipRotation:Boolean;
public var growSpeed:Number;
public
```

```
var hitLocationX:int;
public var hitLocationY:int;
public var hitTarget:String;
public var maxRange:Number;
public var ownerID:String;
public var strength:Number;
public var targetID:String;
public var sourceAttachPoint:String;
public var targetAttachPoint:String;
public var targetScatterX:Number;
public var targetScatterY:Number;
public var visibleHitCounter:int;

public function init( ownerID:String, targetID:String, sourceAttachPoint:String,
targetAttachPoint:String, targetScatterX:Number = 0, targetScatterY:Number = 0,
maxRange:Number = 0, attackHit:Boolean =
true, growSpeed:Number = .2, followShipRotation:Boolean = false ):void
{
this.followShipRotation = followShipRotation;
this.ownerID = ownerID;
this.growSpeed = growSpeed;
this.strength = 0;
this.targetID = targetID;
this.sourceAttachPoint = sourceAttachPoint;
this.targetAttachPoint = targetAttachPoint;
this.targetScatterX = targetScatterX;
this.targetScatterY = targetScatterY;
this.maxRange = maxRange;
this.attackHit = attackHit;
visibleHitCounter = 0;
hitLocationX = hitLocationY = 0;
}
}
}

------------------------------------
File 152: igw\com\game\entity\components\battle\Damage.as
package com.game.entity.components.battle
{
public class Damage
{

public var rotOffset:Number;

public function Damage()
{
}
}
}
------------------------------------
File
```

153: igw\com\game\entity\components\battle\DebuffTray.as
```
package com.game.entity.components.battle
{
import com.Application;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.render.Render;
import com.game.entity.components.shared.render.RenderSprite;
import com.game.entity.components.shared.render.RenderSpriteStarling;
import com.game.entity.components.shared.render.RenderStarling;
import com.model.asset.AssetVO;

import flash.utils.Dictionary;

import org.shared.ObjectPool;


public class DebuffTray
{
private var _animation:Animation;
private var _debuffCounts:Dictionary;
private var _debuffIDs:Dictionary;
private var _debuffs:Vector.<AssetVO>;
private var _fields:Vector.<IRender>;
private var _stackCounts:Dictionary;

public function init():void
{
_debuffCounts = new Dictionary();
_debuffIDs = new Dictionary();
_debuffs = new Vector.<AssetVO>;
_fields = new Vector.<IRender>;
_stackCounts = new Dictionary();
}

public function Draw( anim:Animation, assetVO:AssetVO = null ):void
{
if (!anim)
{
removeIcons();
_animation.removeListener(onAnimationReady);
_animation.alpha = 0;
_animation = anim;
} else
{
_animation = anim;
_animation.label = assetVO.spriteName;
_animation.alpha = 0;
if (!_animation.render)
_animation.addListener(onAnimationReady);
else
```

```
onAnimationReady(Animation.ANIMATION_RENDER_ADDED, anim);
}
}

public function addDebuff( id:String, assetVo:AssetVO, stackCount:int ):void
{
if (id != null)
_debuffIDs[id] = assetVo.type;
_stackCounts[assetVo.type] = stackCount
if (!_debuffCounts.hasOwnProperty(assetVo.type))
{
_debuffCounts[assetVo.type] = 1;
_debuffs.push(assetVo);
if (_animation && _animation.ready && _animation.render)
createDebuffIcon(assetVo, _debuffs.length - 0, stackCount);
}
updateDebuffIcons();
}

private function onAnimationReady( state:int, anim:Animation ):void
{
if (state == Animation.ANIMATION_RENDER_ADDED)
{
_animation.alpha = 1;
_fields.push(anim.render);
for (var i:int = 0; i < _debuffs.length; i++)
{
createDebuffIcon(_debuffs[i], i, _stackCounts[_debuffs[i].type]);
}
updateDebuffIcons();
}
}

private function createDebuffIcon( assetVO:AssetVO, index:int, stackCount:int ):void
{
if (index > 0)
{
var render:IRender = getRender();
if (Application.STARLING_ENABLED)
{
RenderSpriteStarling(_animation.render).addChild(RenderStarling(render));
} else
{
RenderSprite(_animation.render).addChild(Render(render));
}
_fields.push(render);
}
}

private
```

```
function updateDebuffIcons():void
{
if (_animation && _animation.ready && _animation.render)
{
for (var i:int = 0; i < _debuffs.length; i++)
{
var assetVO:AssetVO = _debuffs[i];
if(assetVO == null)
continue;

var spriteName:String = assetVO.spriteName;
var levelPattern:RegExp = /1/i;
var stackCount:int = _stackCounts[assetVO.type];
if (stackCount > 3)
stackCount = 3;
else if(stackCount == 0)
{
//This is required because of the way the debuff sprites are named (They start at 1 instead of 0)
stackCount = 1;
}
spriteName = spriteName.replace(levelPattern, stackCount);

if(_fields.length > i)
_fields[i].updateFrame(_animation.spritePack.getFrame(spriteName, 0), _animation);
}
updateIconPositions();
}
}

private function updateIconPositions():void
{
for (var i:int = 1; i < _fields.length; i++)
{
_fields[i].x = i * 15;
_fields[i].y = 0;
}
}

public function removeDebuff( id:String ):void
{
var count:Number = _debuffCounts[_debuffIDs[id]];
count--;
_debuffCounts[_debuffIDs[id]] = count;
if (count == 0)
{
delete _debuffCounts[_debuffIDs[id]];
delete _debuffCounts[_debuffIDs[id]];
for (var i:int = 0; i < _debuffs.length; i++)
{
if (_debuffs[i].type == _debuffIDs[id])
{
```

```
_debuffs.splice(i, 1);
delete _debuffIDs[id];
if (_fields.length != _debuffs.length && _debuffs.length != 0 && _animation != null)
{
i = 1;
if (Application.STARLING_ENABLED)
RenderSpriteStarling(_animation.render).removeChild(RenderStarling(_fields[i]));
else
RenderSprite(_animation.render).removeChild(Render(_fields[i]));
ObjectPool.give(_fields[i]);
_fields.splice(i, 1);
}
updateDebuffIcons();
updateIconPositions();

return;
}
}
}

}

private function removeIcons():void
{
for (var i:int = 1; i < _fields.length; i++)
{
if(_fields[0] != null && _fields[i] != null)
{
if (Application.STARLING_ENABLED)
RenderSpriteStarling(_fields[0]).removeChild(RenderStarling(_fields[i]));
else
RenderSprite(_fields[0]).removeChild(Render(_fields[i]));
}
if(_fields[i] != null)
ObjectPool.give(_fields[i]);
}
if (_animation)
_animation.removeListener(onAnimationReady);
_fields.length = 0;
}

private function getRender():IRender
{
var render:IRender;
if (Application.STARLING_ENABLED)
{
render = ObjectPool.get(RenderStarling);
} else
{
render
```

```
= ObjectPool.get(Render);
}
return render;
}

public function isDebuffsEmpty():Boolean
{
if (_debuffs.length == 0)
return true;
else
return false;
}

public function destroy():void
{
removeIcons();
if (_animation)
_animation.removeListener(onAnimationReady);
_animation = null;
_debuffCounts = null;
_debuffs.length = 0;
_debuffCounts = null;
_debuffIDs = null;
}




}
}


-------------------------------------
File 154: igw\com\game\entity\components\battle\Drone.as
package com.game.entity.components.battle
{
import org.ash.core.Entity;

public class Drone
{
public var ownerID:String;
public var targetID:String;
public var fireDuration:Number;
public var minWeaponTime:Number;
public var maxWeaponTime:Number;
public var nextFireTick:Number;
public var cleanupTick:Number;
public var currentTick:Number;
public var weaponProto:String;
public var weaponAttack:Entity;
public
```

```
var isOribiting:Boolean;

public function init( ownerID:String, targetID:String ):void
{
this.ownerID = ownerID;
this.targetID = targetID;
this.minWeaponTime = 0.1;
this.maxWeaponTime = 0.5;
this.nextFireTick = -1;
this.cleanupTick = -1;
this.currentTick = -1;
this.weaponProto = "";
this.weaponAttack = null;
this.isOribiting = false;
}

public function destroy():void
{
ownerID = targetID = weaponProto = null;
weaponAttack = null;
}
}
}

-------------------------------------
File 155: igw\com\game\entity\components\battle\Health.as
package com.game.entity.components.battle
{
import com.game.entity.components.shared.Animation;
import com.model.battle.BattleEntityVO;

import org.osflash.signals.Signal;

public class Health
{
public var animation:Animation;

private var _battleEntityVO:BattleEntityVO;
private var _conversion:Number;
private var _currentHealth:int = 0;
private var _damageThreshold:Number;
private var _maxHealth:int;
private var _percent:Number;
private var _signal:Signal;
private var _temp:Number;

public function Health()
{
_signal = new Signal(Number, Number);
}

public
```

```
function init( current:int, max:int, beVO:BattleEntityVO, damageThreshold:Number = -1 ):void
{
if (max <= 0)
max = 1;
_conversion = 1 / max;
_currentHealth = current;
_damageThreshold = damageThreshold;
_percent = _currentHealth * _conversion;
_maxHealth = max;

if (beVO)
{
_battleEntityVO = beVO;
_battleEntityVO.healthPercent = _percent;
}
}

public function addListener( callback:Function ):void { _signal.add(callback); }
public function removeListener( callback:Function ):void { _signal.remove(callback); }

public function get maxHealth():int { return _maxHealth; }
public function set maxHealth( value:int ):void
{
_conversion = 1 / value;
_maxHealth = value;
_temp = percent;
_percent = _currentHealth * _conversion;
_signal.dispatch(_percent, _temp - percent);
}

public function get currentHealth():int { return _currentHealth; }
public function set currentHealth( value:int ):void
{
_temp = _currentHealth;
_currentHealth = value;
_percent = _currentHealth * _conversion;
if (_battleEntityVO)
_battleEntityVO.healthPercent = _percent;
_signal.dispatch(_percent, _temp - _currentHealth);
}

public function get damageThreshold():Number { return _damageThreshold; }
public function set damageThreshold( v:Number ):void { _damageThreshold = v; }

public function get percent():Number { return _percent; }

public function destroy():void
{
animation = null;
if (_battleEntityVO)
_battleEntityVO.healthPercent
```

```
= 0;
_battleEntityVO = null;
_damageThreshold = -1;
_signal.removeAll();
}
}
}

------------------------------------
File 156: igw\com\game\entity\components\battle\Modules.as
package com.game.entity.components.battle
{
import com.model.prototype.IPrototype;

import flash.utils.Dictionary;
import flash.utils.getTimer;

import org.ash.core.Entity;

public class Modules
{
private var _activatedModules:Array = [];
private var _activatedTimes:Array = [];
private var _modules:Array = [];
private var _entityModules:Array = [];
private var _indiciesByAttachPoint:Dictionary = new Dictionary();
private var _modulesByAttachPoint:Dictionary = new Dictionary();

public var moduleStates:Dictionary = new Dictionary();

public function addActivatedModule( id:int, activatedPrototypeVO:IPrototype ):void
{
_activatedModules[id] = activatedPrototypeVO;
}

public function activateModule( id:int ):void
{
if (_activatedModules[id] != null)
{
_activatedTimes[id] = getTimer();
}
}

public function isActive( id:int ):Boolean
{
if (_activatedModules[id] != null)
{
if (_activatedTimes[id] == null || moduleCooldownTimeRemaining(id) == 0)
return true;
}
return
```

```
false;
}

public function moduleCooldownTimeRemaining( id:int ):Number
{
if (_activatedModules[id] != null)
{
if (_activatedTimes[id] != null)
{
var t:Number = getTimer() - _activatedTimes[id];
var remaining:Number = getModuleReloadTime(id) - t;
if (remaining <= 0)
{
_activatedTimes[id] = null;
remaining = 0;
}
return remaining;
}
}
return 0;
}

public function getModuleReloadTime( id:int ):Number
{
if (_activatedModules[id] != null)
return Number(_activatedModules[id].getValue('reloadTime')) * 1000;
return 0;
}

public function addModule( id:int, prototypeVO:IPrototype ):void
{
_modules[id] = prototypeVO;
}

public function addModuleByAttachPoint( attachPoint:String, prototypeVO:IPrototype ):void {
_modulesByAttachPoint[attachPoint] = prototypeVO; }

public function getModuleByAttachPoint( attachPoint:String ):IPrototype { return
_modulesByAttachPoint[attachPoint]; }

public function addIndexByAttachPoint( attachPoint:String, moduleIndex:Number ):void {
_indiciesByAttachPoint[attachPoint] = moduleIndex; }

public function getModuleIndexByAttachPoint( attachPoint:String ):Number { return
_indiciesByAttachPoint[attachPoint]; }

public function getModule( id:int ):IPrototype
{
return _modules[id];
}

public
```

```
function addEntityModule( id:int, defense:Entity ):void
{
if (defense != null)
_entityModules[id] = defense;
}

public function getEntityModule( id:int ):Entity
{
return _entityModules[id];
}

public function get activatedModules():Array { return _activatedModules; }

public function get entityModules():Array { return _entityModules; }

public function destroy():void
{
_activatedModules.length = 0;
_activatedTimes.length = 0;
_modules.length = 0;
_entityModules.length = 0;
_indiciesByAttachPoint = new Dictionary();
_modulesByAttachPoint = new Dictionary();
moduleStates = new Dictionary();
}
}
}
```

File 157: igw\com\game\entity\components\battle\Shield.as

```
package com.game.entity.components.battle
{
import com.game.entity.components.shared.Animation;

import org.osflash.signals.Signal;

public class Shield
{
public var animation:Animation;

private var _currentStrength:int;
private var _enabled:Boolean;
private var _enabledSignal:Signal;
private var _isBuildingShield:Boolean;
private var _strengthSignal:Signal;

public function Shield()
{
_enabledSignal = new Signal(Boolean);
_strengthSignal = new Signal(int);
}
```

```
public function init( enabled:Boolean, health:int, forBuilding:Boolean = false ):void
{
_enabled = enabled;
_currentStrength = health;
_isBuildingShield = forBuilding;
}

public function addStrengthListener( callback:Function ):void { _strengthSignal.add(callback); }
public function removeStrengthListener( callback:Function ):void {
_strengthSignal.remove(callback); }

public function addEnableListener( callback:Function ):void { _enabledSignal.add(callback); }
public function removeEnableListener( callback:Function ):void {
_enabledSignal.remove(callback); }

public function get enabled():Boolean { return _enabled; }
public function set enabled( value:Boolean ):void
{
if (value != _enabled)
{
_enabled = value;
_enabledSignal.dispatch(_enabled);
}
}

public function get currentStrength():int { return _currentStrength; }
public function set currentStrength( value:int ):void
{
if (_currentStrength != value)
{
_currentStrength = value;
_strengthSignal.dispatch(_currentStrength);
}
}

public function get isBuildingShield():Boolean { return _isBuildingShield; }

public function destroy():void
{
_enabledSignal.removeAll();
_strengthSignal.removeAll();
_isBuildingShield = false;
}
}
}
```

File 158: igw\com\game\entity\components\battle\Ship.as

```
package
```

```
com.game.entity.components.battle
{
import flash.geom.Point;

import org.ash.core.Entity;

public class Ship
{
public var attachments:Vector.<Entity>;
public var position1:Point;
public var position2:Point;
public var position3:Point;
public var lastUpdate:Number;
public var rangeReference:String;
public var thrustersFront:Boolean;
public var thrustersRight:Boolean;
public var thrustersBack:Boolean;
public var thrustersLeft:Boolean;
public var accelThreshold:Number;

public var damageState:Number;
public var damageEffects:Vector.<Entity>;

public function Ship()
{
thrustersFront = thrustersRight = thrustersBack = thrustersLeft = false;
attachments = new Vector.<Entity>();
lastUpdate = 0;
position1 = new Point();
position2 = new Point();
position3 = new Point();
accelThreshold = 0.1;

damageState = 0;
damageEffects = new Vector.<Entity>();
}

public function destroy( final:Boolean = true ):void
{
accelThreshold = 0.1;
lastUpdate = 0;
position1.setTo(0, 0);
position2.setTo(0, 0);
position3.setTo(0, 0);
attachments.length = 0;
thrustersFront = thrustersRight = thrustersBack = thrustersLeft = false;

damageState = 0;
damageEffects.length = 0;
if (final)
rangeReference
```

```
= null;
}
}
}
```

File 159: igw\com\game\entity\components\battle\TrailFX.as

```
package com.game.entity.components.battle
{
import flash.geom.Point;

import org.ash.core.Entity;

public class TrailFX
{
// Configuration members
public var alphaChange:Number;
public var color:uint;
public var maxSegments:int;
public var type:String;
public var thickness:Number;

// Runtime data tracking
public var currentSegment:Entity;
public var segments:Vector.<Entity>;

public var lastPosition:Point = new Point();

public function TrailFX()
{
segments = new Vector.<Entity>();
}

public function destroy():void
{
currentSegment = null;
lastPosition.setTo(0, 0);
segments.length = 0;
}
}
}
```

File 160: igw\com\game\entity\components\sector\Fleet.as

```
package com.game.entity.components.sector
{
import org.ash.core.Entity;

public
```

```
class Fleet
{
public var thrusterBackLeft:Entity;
public var thrusterBackRight:Entity;
public var thrustersEngaged:Boolean;

public function Fleet()
{
thrustersEngaged = false;
}

public function disengageThrusters():void
{
thrusterBackLeft = thrusterBackRight = null;
thrustersEngaged = false;
}

public function destroy():void
{
disengageThrusters();
}
}
}

-------------------------------------
File 161: igw\com\game\entity\components\sector\Mission.as
package com.game.entity.components.sector
{
public class Mission
{
public function Mission()
{
}
}
}
-------------------------------------
File 162: igw\com\game\entity\components\sector\Transgate.as
package com.game.entity.components.sector
{
public class Transgate
{
public var isPositiveWarp:Boolean;
public var customDestinationPrototypeGroup:String;

public function destroy():void
{

}
}
}

-------------------------------------
```

File 163: igw\com\game\entity\components\shared\Animation.as

```
package com.game.entity.components.shared
{
import com.Application;
import com.model.asset.ISpritePack;

import org.osflash.signals.Signal;
import org.starling.textures.Texture;

public class Animation
{
public static const ANIMATION_READY:int = 0;
public static const ANIMATION_RENDER_ADDED:int = 1;
public static const ANIMATION_COMPLETE:int = 2;

public var allowTransform:Boolean;
public var blendMode:String;
public var center:Boolean;
public var color:uint = 0;
public var destroyOnComplete:Boolean;
public var frame:int;
public var labelChanged:Boolean;
public var numberOfFrames:int;
public var offsetX:Number;
public var offsetY:Number;
public var playing:Boolean;
public var replay:Boolean;
public var scaleX:Number;
public var scaleY:Number;
public var spritePack:ISpritePack;
public var textChanged:Boolean;
public var textLostContext:Boolean;
public var time:Number;
public var transformScaleFirst:Boolean;
public var visible:Boolean;
public var duration:Number;
public var randomStart:Boolean;

private var _alpha:Number;
private var _fps:Number;
private var _frameDuration:Number;
private var _label:String;
private var _lostContext:Boolean;
private var _numListeners:int;
private var _ready:Boolean;
private var _render:IRender;
private var _signal:Signal;
private var _sprite:*;
private var _text:String;
private
```

```
var _type:String;

public function init( type:String, label:String, center:Boolean = false, frame:int = 0, fps:Number =
30, visible:Boolean = false, xoffset:Number = 0, yoffset:Number = 0 ):void
{
this.center = center;
allowTransform = destroyOnComplete = labelChanged = _lostContext = _ready = textChanged
= textLostContext = false;
this.frame = frame;
duration = -1.0;
_label = label;
_type = type;
offsetX = xoffset;
offsetY = yoffset;
playing = replay = transformScaleFirst = true;
time = 0;
_alpha = numberOfFrames = scaleX = scaleY = 1;
_fps = fps;
_frameDuration = 1 / _fps;
_numListeners = 0;
this.visible = visible;
spritePack = null;
randomStart = false;
}

public function addListener( callback:Function ):void
{
if (!_signal)
_signal = new Signal(int, Animation);
_numListeners++;
_signal.add(callback);
}

public function removeListener( callback:Function ):void
{
if (_signal)
{
_signal.remove(callback);
_numListeners--;
}
}

public function dispatch( type:int ):void
{
if (_numListeners > 0)
_signal.dispatch(type, this);
}

public function forceReady( v:Boolean = true ):void { _ready = v; }
public function deviceLostContext():void
{
```

```
if (center && _sprite)
{
if (Application.STARLING_ENABLED)
{
if (render)
{
render.x += Texture(_sprite).frame.width * .5;
render.y += Texture(_sprite).frame.height * .5;
}
offsetX -= Texture(_sprite).frame.width * .5;
offsetY -= Texture(_sprite).frame.height * .5;
} else
{
offsetX -= _sprite.width * .5;
offsetY -= _sprite.height * .5;
}
}
_lostContext = true;
_ready = false;
if (_text)
{
textChanged = textLostContext = true;
if (spritePack == null)
_ready = true;
}
spritePack = null;
_sprite = null;
}

public function get alpha():Number { return _alpha; }
public function set alpha( v:Number ):void { _alpha = v; if (render) render.alpha = _alpha; }

public function get lostContext():Boolean { return _lostContext; }

public function set forceSprite( v:* ):void { _sprite = v; }
public function get sprite():* { return _sprite; }
public function set sprite( v:* ):void
{
if (v)
{
if (!_sprite)
{
if (center)
{
if (Application.STARLING_ENABLED)
{
offsetX += Texture(v).frame.width * .5;
offsetY += Texture(v).frame.height * .5;
} else
{
```

```
offsetX += v.width * .5;
offsetY += v.height * .5;
}
}
_lostContext = false;
_ready = true;
}
_sprite = v;
}
}

public function get label():String { return _label; }
public function set label( v:String ):void { if (_label != v) labelChanged = true; _label = v; }
[Inline]
public function get ready():Boolean { return _ready; }
public function get render():IRender { return _render; }
public function set render( v:IRender ):void { _render = v; if (_render)
dispatch(ANIMATION_RENDER_ADDED); }
public function get text():String { return _text; }
public function set text( v:String ):void { _text = v; textChanged = true; }
public function get type():String { return _type; }
public function set type( v:String ):void
{
_type = v;
deviceLostContext();
_lostContext = false;
frame = 0;
}

public function get height():Number
{
if (_sprite)
{
if (Application.STARLING_ENABLED)
return Texture(_sprite).frame.height;
return _sprite.height;
}
return 0;
}
public function get width():Number
{
if (_sprite)
{
if (Application.STARLING_ENABLED)
return Texture(_sprite).frame.width;
return _sprite.width;
}
return 0;
}

public
```

```
function get frameDuration():Number
{
// By default return the standard value
if (this.duration < 0.0)
return _frameDuration;

// If there's a duration scale to that length
return (this.duration / numberOfFrames);
}

public function destroy():void
{
blendMode = null;
color = 0;
render = null;
spritePack = null;
if (_signal)
_signal.removeAll();
_signal = null;
_sprite = null;
_text = null;
}
}
}

------------------------------------
File 164: igw\com\game\entity\components\shared\Cargo.as
package com.game.entity.components.shared
{
public class Cargo
{
public var cargo:Number;
}
}

------------------------------------
File 165: igw\com\game\entity\components\shared\DebugLine.as
package com.game.entity.components.shared
{
public class DebugLine
{
public var startColor:uint;
public var endColor:uint;
}
}

------------------------------------
File 166: igw\com\game\entity\components\shared\Detail.as
package com.game.entity.components.shared
{
import
```

```
com.model.asset.AssetVO;
import com.model.prototype.IPrototype;

public class Detail
{
public var assetVO:AssetVO;
public var category:String;
public var level:int;
public var ownerID:String;
public var prototypeVO:IPrototype;
public var baseLevel:uint;
public var waypointType:String;
public var baseRatingTech:uint;
public var maxPlayersPerFaction:uint;

public function init( category:String, asset:AssetVO, prototype:IPrototype = null, ownerID:String
= '', baseLevel:uint = 0, waypointType:String = '', baseRatingTech:uint = 0 ):void
{
this.category = category;
this.assetVO = asset;
this.prototypeVO = prototype;
this.ownerID = ownerID;
this.baseLevel = baseLevel;
this.waypointType = waypointType;
this.baseRatingTech = baseRatingTech;
this.maxPlayersPerFaction = 0;
}

public function get spriteName():String { return assetVO ? assetVO.spriteName : ''; }
public function get type():String { return assetVO ? assetVO.type : ''; }

public function destroy():void
{
assetVO = null;
prototypeVO = null;
ownerID = '';
}
}
}

------------------------------------
File 167: igw\com\game\entity\components\shared\Enemy.as
package com.game.entity.components.shared
{
public class Enemy
{
public function Enemy()
{
}
}
}
```

File 168: igw\com\game\entity\components\shared\EventComponent.as
```
package com.game.entity.components.shared
{
public class EventComponent
{
public function EventComponent()
{
}
}
}
```

File 169: igw\com\game\entity\components\shared\Grid.as
```
package com.game.entity.components.shared
{
public class Grid
{
public var hashTL:int;
public var hashBR:int;
}
}
```

File 170: igw\com\game\entity\components\shared\Interactable.as
```
package com.game.entity.components.shared
{
public class Interactable
{
public var selected:Boolean = false;

public function destroy():void
{
selected = false;
}
}
}
```

File 171: igw\com\game\entity\components\shared\IRender.as
```
package com.game.entity.components.shared
{
public interface IRender
{
/**
*
* @param image
* @param animation
* @param forceResize Only applicable in starling but forces an image to resize to fit a texture
*
*/
```

```
function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void;

function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void;

function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number = .5
):void;
function removeGlow():void;

function get alpha():Number;
function set alpha( v:Number ):void;

function get blendMode():String;
function set blendMode( v:String ):void;

function get color():uint;
function set color( value:uint ):void;

function get rotation():Number;
function set rotation( v:Number ):void;

function get width():Number;
function get height():Number;

function set width( v:Number ):void;
function set height( v:Number ):void;

function get scaleX():Number;
function get scaleY():Number;

function set scaleX( v:Number ):void;
function set scaleY( v:Number ):void;

function get x():Number;
function get y():Number;

function set x( v:Number ):void;
function set y( v:Number ):void;

function destroy():void;
}
}

-----------------------------------
File 172: igw\com\game\entity\components\shared\Move.as
package com.game.entity.components.shared
{
import com.controller.ServerController;

import
```

```
flash.geom.Point;

public class Move
{
public var delta:Point = new Point();
public var destination:Point = new Point();
public var destroyOnComplete:Boolean;
public var directionChanged:Boolean;
public var endTick:int;
public var lerping:Boolean;
public var lerpDestination:Point = new Point();
public var moving:Boolean;
public var relative:Boolean;
public var rotation:Number;
public var start:Point = new Point();
public var startTick:int;
public var time:Number;
public var totalTime:Number;
public var type:int;
public var velocity:Point = new Point();
public var fadeOut:int;

private var _updates:Array = [];

public function init( speed:int, type:int ):void
{
delta.setTo(0, 0);
destination.setTo(0, 0);
lerpDestination.setTo(0, 0);
start.setTo(0, 0);
endTick = time = startTick = totalTime = 0;
destroyOnComplete = directionChanged = lerping = moving = relative = false;
fadeOut = 0;
this.type = type;
}

public function setDelta( x:Number, y:Number ):void
{
delta.x = x;
delta.y = y;
}

public function setLerpDestination( x:Number, y:Number, velocityX:Number, velocityY:Number,
rotation:Number, startTick:int, endTick:int ):void
{
lerpDestination.x = x;
lerpDestination.y = y;
this.rotation = rotation;
directionChanged = true;
this.endTick = endTick;
this.startTick
```

```
= startTick;
time = (startTick < ServerController.SIMULATED_TICK) ?
((Math.min(ServerController.SIMULATED_TICK, endTick) - startTick) * .1) : 0; //(moving && type
== EntityMoveEnum.FREE) ? time -= totalTime : 0;
totalTime = (endTick - startTick) * .1;
if (totalTime == 0)
totalTime = .1;
velocity.setTo(velocityX, velocityY);
moving = true;
}

public function setPointToPoint( x:Number, y:Number, startTick:int, endTick:int, relative:Boolean
= false ):void
{
setDestination(x, y);
directionChanged = true;
this.endTick = endTick;
moving = true;
this.relative = relative;
this.startTick = startTick;
time = (startTick < ServerController.SIMULATED_TICK) ?
((Math.min(ServerController.SIMULATED_TICK, endTick) - startTick) * .1) : 0;
totalTime = (endTick - startTick) * .1;
if (totalTime == 0)
totalTime = .1;
}

public function setDestination( x:Number, y:Number ):void
{
destination.x = x;
destination.y = y;
}

public function setStart( position:Point ):void
{
start.x = position.x;
start.y = position.y;
}

public function addUpdate( targetX:Number, targetY:Number, velocityX:Number,
velocityY:Number, rotation:Number, startTick:int, endTick:int ):void
{
_updates.push(targetX, targetY, velocityX, velocityY, rotation, startTick, endTick);
}

public function setNextUpdate():void
{
setLerpDestination(_updates[0], _updates[1], _updates[2], _updates[3], _updates[4],
_updates[5], _updates[6]);
_updates.splice(0, 7);
lerping
```

```
= true;
}

public function get hasUpdate():Boolean { return _updates.length > 0; }

public function destroy():void
{
_updates.length = 0;
}
}
}
```

File 173: igw\com\game\entity\components\shared\Muzzle.as
```
package com.game.entity.components.shared
{
public class Muzzle
{
public var moduleIdx:int;
public var currentFrame:Number;
public var chargeDuration:Number;
public var baseScale:Number;
public var charging:Boolean;
public var weaponClass:String;

public function Muzzle()
{
}
}
}
```

File 174: igw\com\game\entity\components\shared\Owned.as
```
package com.game.entity.components.shared
{
public class Owned
{
public function Owned()
{
}
}
}
```

File 175: igw\com\game\entity\components\shared\Position.as
```
package com.game.entity.components.shared
{
import flash.geom.Point;

import org.shared.ObjectPool;

public class Position
{
```

```
public var depthDirty:Boolean;
public var dirty:Boolean;
public var ignoreRotation:Boolean = false;
public var linkedTo:Position;
public var parallaxSpeed:Number;
public var position:Point = new Point();

private var _depth:int;
private var _layer:int = 0;
private var _links:Vector.<Position>;
private var _oldLayer:int = -1;
private var _rotation:Number = 0;
private var _rotationDelta:Number = 0;
private var _startRotation:Number = 0;
private var _targetRotation:Number = 0;

public function Position()
{
_links = new Vector.<Position>;
}

public function init( x:Number, y:Number, rotation:Number, layer:int = 10, parallaxSpeed:Number
= 1 ):void
{
this.rotation = rotation;
this.targetRotation = rotation;
_layer = layer;
this.parallaxSpeed = parallaxSpeed;
dirty = true;
_depth = -1;
depthDirty = false;
position.setTo(x, y);
}

public function addLink( pos:Position ):void
{
pos.linkedTo = this;
_links.push(pos);
}

public function removeLink( pos:Position ):void
{
var index:int = _links.indexOf(pos);
if (index > -1)
_links.splice(index, 1);
}

public function clearRotation():void
{
_rotation
```

```
= 0;
_startRotation = 0;
_rotationDelta = 0;
_targetRotation = 0;
}

public function clone():Position
{
var pos:Position = ObjectPool.get(Position);
pos.init(position.x, position.y, rotation, layer, parallaxSpeed);
return pos;
}

public function layerSwap( oldLayer:int, newLayer:int ):void
{
_oldLayer = oldLayer;
_layer = newLayer;
depthDirty = true;
}

public function get depth():int { return _depth; }
public function set depth( v:int ):void
{
if (_links.length > 0)
{
var diff:int;
for (var i:int = 0; i < _links.length; i++)
{
diff = _links[i].depth - _depth;
_links[i].depth = v + diff;
}
}
_depth = v;
depthDirty = true;
}

public function get layer():int
{
if (_oldLayer > -1)
{
var l:int = _oldLayer;
_oldLayer = -1;
return l;
}
return _layer;
}
public function set layer( v:int ):void { _layer = v; }

public function get rotation():Number { return _rotation; }
public function set rotation( v:Number ):void
{
```

```
if (!ignoreRotation)
{
_rotation = v;
dirty = true;
if (_links.length > 0)
{
for (var i:int = 0; i < _links.length; i++)
_links[i].rotation = v;
}
}
}

public function get rotationDelta():Number { return _rotationDelta; }
public function get startRotation():Number { return _startRotation; }
public function get targetRotation():Number { return _targetRotation; }
public function set targetRotation( v:Number ):void
{
_startRotation = _rotation;
_rotationDelta = v - _startRotation;
_rotationDelta = Math.atan2(Math.sin(_rotationDelta), Math.cos(_rotationDelta));
_targetRotation = _startRotation + _rotationDelta;

dirty = true;
if (_links.length > 0)
{
for (var i:int = 0; i < _links.length; i++)
_links[i]._targetRotation = v;
}
}

public function get x():Number { return position.x; }
public function set x( v:Number ):void
{
position.x = v;
dirty = true;
if (_links.length > 0)
{
for (var i:int = 0; i < _links.length; i++)
_links[i].x = v;
}
}

public function get y():Number { return position.y; }
public function set y( v:Number ):void
{
position.y = v;
dirty = true;
if (_links.length > 0)
{
for
```

```
(var i:int = 0; i < _links.length; i++)
_links[i].y = v;
}
}

public function destroy():void
{
ignoreRotation = false;
if (linkedTo)
linkedTo.removeLink(this);
linkedTo = null;
_links.length = 0;
}
}
}

-------------------------------------
File 176: igw\com\game\entity\components\shared\Pylon.as
package com.game.entity.components.shared
{
import com.game.entity.nodes.starbase.BuildingNode;

import org.ash.core.Entity;

public class Pylon
{
private static const FIELDS:Object = {};

public var baseX:int;
public var baseY:int;
public var bottom:Entity;
public var color:uint;
public var node:BuildingNode;

private var _bottomConnection:BuildingNode;
private var _leftConnection:BuildingNode;
private var _rightConnection:BuildingNode;
private var _topConnection:BuildingNode;

public function addConnection( connection:BuildingNode ):Object
{
var temp:BuildingNode;
if (connection.$pylon.baseX == baseX)
{
if (connection.$pylon.baseY < baseY)
{
if (!_topConnection)
{
_topConnection = connection;
return craftReturn(connection, null);
}
```

```
else if (connection.building.buildingVO.baseY > _topConnection.building.buildingVO.baseY ||
connection == _topConnection)
{
if (connection != _topConnection)
{
temp = _topConnection;
removeConnection(_topConnection);
}
_topConnection = connection;
return craftReturn(connection, temp);
}
} else
{
if (!_bottomConnection)
{
_bottomConnection = connection;
return craftReturn(connection, null);
} else if (_bottomConnection && connection.building.buildingVO.baseY <
_bottomConnection.building.buildingVO.baseY || connection == _bottomConnection)
{
if (connection != _bottomConnection)
{
temp = _bottomConnection;
removeConnection(_bottomConnection);
}
_bottomConnection = connection;
return craftReturn(connection, temp);
}
}
} else if (connection.$pylon.baseY == baseY)
{
if (connection.$pylon.baseX < baseX)
{
if (!_leftConnection)
{
_leftConnection = connection
return craftReturn(connection, null);
} else if (_leftConnection && connection.building.buildingVO.baseX >
_leftConnection.building.buildingVO.baseX || connection == _leftConnection)
{
if (connection != _leftConnection)
{
temp = _leftConnection;
removeConnection(_leftConnection);
}
_leftConnection = connection;
return craftReturn(connection, temp);
}
} else
{
if
```

```
(!_rightConnection)
{
_rightConnection = connection;
return craftReturn(connection, null);
} else if (_rightConnection && connection.building.buildingVO.baseX <
_rightConnection.building.buildingVO.baseX || connection == _rightConnection)
{
if (connection != _rightConnection)
{
temp = _rightConnection;
removeConnection(_rightConnection);
}
_rightConnection = connection;
return craftReturn(connection, temp);
}
}
}
return null;
}

public function removeConnection( connection:BuildingNode, notifyConnection:Boolean = true
):void
{
if (_bottomConnection == connection)
_bottomConnection = null;
if (_leftConnection == connection)
_leftConnection = null;
if (_rightConnection == connection)
_rightConnection = null;
if (_topConnection == connection)
_topConnection = null;
if (notifyConnection)
connection.$pylon.removeConnection(node, false);
}

public function clearConnections():void
{
_bottomConnection = _leftConnection = _rightConnection = _topConnection = null;
}

public function craftKey( bnode:BuildingNode ):String
{
if (bnode == null)
return null;
return "Forcefield" + ((node.entity.id < bnode.entity.id) ? node.entity.id + "-" + bnode.entity.id :
bnode.entity.id + "-" + node.entity.id);
}

private function craftReturn( added:BuildingNode, removed:BuildingNode ):Object
{
FIELDS.added
```

```
= craftKey(added);
FIELDS.removed = craftKey(removed);
return FIELDS;
}

public function get bottomConnection():BuildingNode { return _bottomConnection; }
public function get bottomWallKey():String { return craftKey(_bottomConnection); }

public function get leftConnection():BuildingNode { return _leftConnection; }
public function get leftWallKey():String { return craftKey(_leftConnection); }

public function get rightConnection():BuildingNode { return _rightConnection; }
public function get rightWallKey():String { return craftKey(_rightConnection); }

public function get topConnection():BuildingNode { return _topConnection; }
public function get topWallKey():String { return craftKey(_topConnection); }

public function destroy():void
{
node = null;
_bottomConnection = null;
_leftConnection = null;
_rightConnection = null;
_topConnection = null;
}
}
}

------------------------------------
File 177: igw\com\game\entity\components\shared\Thruster.as
package com.game.entity.components.shared
{
public class Thruster
{
public var direction:String;

public function Thruster()
{
}
}
}
------------------------------------
File 178: igw\com\game\entity\components\shared\VCList.as
package com.game.entity.components.shared
{
import org.ash.core.Entity;

public class VCList
{
private var _addCallback:Function;
private
```

```
var _entity:Entity;
private var _components:Vector.<Entity> = new Vector.<Entity>;
private var _names:Array;
private var _removeCallback:Function;
private var _updateCallback:Function;

public function init( ... componentNames ):void
{
_names = (componentNames) ? componentNames : [];
}

public function addComponentType( type:String ):void
{
var index:int = _names.indexOf(type);
if (index == -1)
{
_names.push(type);
_addCallback && _addCallback(_entity, type);
} else
_updateCallback && _updateCallback(_entity, type);
}

public function hasComponentType( type:String ):Boolean { return _names.indexOf(type) > -1 ?
true : false; }

public function removeComponentType( type:String ):void
{
if (!_names)
return;
var index:int = _names.indexOf(type);
if (index > -1)
{
_names.splice(index, 1);
_removeCallback && _removeCallback(_entity, type);
}
}

public function addComponent( component:Entity ):void
{
_components.push(component);
}

public function getComponent( type:String ):Entity
{
for (var i:int = 0; i < _components.length; i++)
{
if (Detail(_components[i].get(Detail)).type == type)
return _components[i];
}
return null;
}
```

```
public function removeComponent( component:Entity ):void
{
var index:int = _components.indexOf(component);
if (index > -1)
_components.splice(index, 1);
}

public function addCallbacks( add:Function, update:Function, remove:Function, entity:Entity
):void
{
_addCallback = add;
_entity = entity;
_removeCallback = remove;
_updateCallback = update;
}

public function removeCallbacks():void
{
_addCallback = _removeCallback = _updateCallback = null;
_entity = null;
}

public function get components():Vector.<Entity> { return _components; }
public function get names():Array { return _names; }

public function destroy():void
{
_components.length = 0;
_names = null;
removeCallbacks();
}
}
}

------------------------------------
File 179: igw\com\game\entity\components\shared\fsm\BuildingShieldFSM.as
package com.game.entity.components.shared.fsm
{
import com.game.entity.components.shared.Animation;

import org.ash.core.Node;

public class BuildingShieldFSM implements IFSMComponent
{
public static const BEGIN:int = 0;
public static const POWER_ON:int = 1;
public static const STABLE:int = 2;
public static const HIT:int = 3;
public
```

```
static const POWER_OFF:int = 4;
public static const END:int = 5;

public var animation:Animation;

private var _state:int;

public function BuildingShieldFSM()
{
_state = BEGIN;
}

public function advanceState( node:Node ):Boolean
{
var animation:Animation;
switch (_state)
{
case BEGIN:
break;
case POWER_ON:
break;
case STABLE:
break;
case HIT:
break;
case POWER_OFF:
break;
case END:
break;
}
return true;
}

public function get component():IFSMComponent { return this; }

public function get state():int { return _state; }
public function set state( v:int ):void { _state = v; }

public function destroy():void
{
animation = null;
state = BEGIN;
}
}
}

-----------------------------------
File 180: igw\com\game\entity\components\shared\fsm\Forcefield.as
package com.game.entity.components.shared.fsm
{
import
```

```
com.Application;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.render.Render;
import com.game.entity.components.shared.render.RenderSprite;
import com.game.entity.components.shared.render.RenderSpriteStarling;
import com.game.entity.components.shared.render.RenderStarling;
import com.model.starbase.BuildingVO;

import org.ash.core.Node;
import org.shared.ObjectPool;

public class Forcefield implements IFSMComponent
{
public static const BEGIN:int = 0;
public static const POWER_ON:int = 1;
public static const STABLE:int = 2;
public static const POWER_OFF:int = 3;
public static const END:int = 4;

private static const POWER_ON_SPEED:Number = .3;
private static const POWER_OFF_SPEED:Number = .45;

public var animation:Animation;
public var building:BuildingVO;
public var color:uint;

private var _fields:Vector.<IRender>;
private var _render:IRender;
private var _state:int;

public function Forcefield()
{
_fields = new Vector.<IRender>;
_state = BEGIN;
}

public function advanceState( node:Node ):Boolean
{
var i:int;
var lengths:int;
var num:Number;
switch (_state)
{
case BEGIN:
if (animation.ready && animation.render && _fields.length == 0)
{
animation.render.color = color;
animation.render.alpha = 0;
animation.alpha = 0;
_fields.push(animation.render);
```

```
_render = animation.render;
var dir:int = (building.sizeX > 5) ? 1 : -1;
lengths = ((building.sizeX > 5) ? building.sizeX / 5 : building.sizeY / 5) - 1;
for (i = 0; i < lengths; i++)
{
createFieldLength(i + 1, dir);
}
state = POWER_ON;
}
break;

case POWER_ON:
if (!animation.render)
{
removeFieldLengths();
state = BEGIN;
}
lengths = _fields.length - 1;

if(lengths<0)
break;

num = Math.round(_fields.length * .5);

if(_fields.length <= num)
num = _fields.length - 1;

for (i = 0; i < num; i++)
{
if (_fields[i].alpha != 1)
break;
}
if (i == 0)
{
animation.alpha += POWER_ON_SPEED;
if (animation.alpha > 1)
animation.alpha = 1;
}
_fields[i].alpha += POWER_ON_SPEED;
if (_fields[i].alpha >= 1)
{
_fields[i].alpha = 1;
if (i == num - 1)
state = STABLE;
}

if (_fields[i] != _fields[lengths - i])
_fields[lengths - i].alpha = _fields[i].alpha;
break;

case
```

```
STABLE:
if (!animation.render)
{
removeFieldLengths();
state = BEGIN;
}
break;

case POWER_OFF:
removeFieldLengths();
state = BEGIN;
break;

case END:
removeFieldLengths();
state = BEGIN;
return false;
break;
}

return true;
}

public function adjustFieldLengths():void
{
if (_render && _state != BEGIN && _state != POWER_OFF && _state != END)
{
var dir:int = (building.sizeX > 5) ? 1 : -1;
var lengths:int = ((building.sizeX > 5) ? building.sizeX / 5 : building.sizeY / 5);
if (lengths != _fields.length)
{
var lengthRender:IRender;
if (_state == POWER_ON)
{
for (var i:int = 0; i < _fields.length; i++)
_fields[i].alpha = 1;
animation.alpha = 1;
state = STABLE;
}
while (_fields.length != lengths)
{
if (_fields.length < lengths)
createFieldLength(_fields.length, dir, 1);
else
{
lengthRender = _fields.pop();
if (Application.STARLING_ENABLED)
RenderSpriteStarling(_render).removeChild(RenderStarling(lengthRender));
else
RenderSprite(_render).removeChild(Render(lengthRender));
ObjectPool.give(lengthRender);
```

```
}
}
}
}
}

private function createFieldLength( count:int, direction:int, alpha:Number = 0 ):IRender
{
var render:IRender;
if (Application.STARLING_ENABLED)
{
render = ObjectPool.get(RenderStarling);
RenderSpriteStarling(animation.render).addChild(RenderStarling(render));
} else
{
render = ObjectPool.get(Render);
RenderSprite(animation.render).addChild(Render(render));
}
render.x = 64.7 * count * direction;
render.y = 32 * count;
render.alpha = alpha;
render.color = color;
render.updateFrame(animation.spritePack.getFrame(animation.label, 0), animation);
_fields.push(render);
return render;
}

private function removeFieldLengths():void
{
if (_render)
{
for (var i:int = 1; i < _fields.length; i++)
{
if (Application.STARLING_ENABLED)
RenderSpriteStarling(_render).removeChild(RenderStarling(_fields[i]));
else
RenderSprite(_render).removeChild(Render(_fields[i]));
ObjectPool.give(_fields[i]);
}
_render = null;
}
_fields.length = 0;
}

public function get component():IFSMComponent { return this; }

public function get state():int { return _state; }
public function set state( v:int ):void { _state = v; }

public
```

```
function destroy():void
{
removeFieldLengths();
_render = null;
_state = BEGIN;

animation = null;
building = null;
}
}
}
```

File 181: igw\com\game\entity\components\shared\fsm\FSM.as

```
package com.game.entity.components.shared.fsm
{
import org.ash.core.Node;
import org.shared.ObjectPool;

public class FSM implements IFSMComponent
{
private var _component:IFSMComponent;

public function init( component:IFSMComponent ):void
{
_component = component;
}

public function advanceState( node:Node ):Boolean
{
return _component.advanceState(node);
}

public function get component():IFSMComponent { return _component; }

public function get state():int { return _component.state; }
public function set state( v:int ):void { _component.state = v; }

public function destroy():void
{
ObjectPool.give(_component);
_component = null;
}
}
}
```

File 182: igw\com\game\entity\components\shared\fsm\IFSMComponent.as

```
package com.game.entity.components.shared.fsm
{
import
```

```
org.ash.core.Node;

public interface IFSMComponent
{
function advanceState( node:Node ):Boolean;

function get component():IFSMComponent;

function get state():int;
function set state( v:int ):void;

function destroy():void
}
}
```

File 183: igw\com\game\entity\components\shared\fsm\TurretFSM.as

```
package com.game.entity.components.shared.fsm
{
import com.game.entity.components.shared.Animation;
import com.game.entity.nodes.shared.FSMNode;

import flash.events.TimerEvent;
import flash.utils.Timer;

import org.ash.core.Node;

public class TurretFSM implements IFSMComponent
{

public static const DEFAULT:int = -1;
public static const INIT:int = 0;
public static const SCANNING:int = 1;
public static const TRACKING:int = 2;
public static const PAUSED:int = 3;

//the animation component of the turret. If this is null it either means the player does not have a gun equipped or the turret is offscreen
public var animation:Animation;

private var _state:int;

public function TurretFSM()
{
_state = INIT;
}

private var crntRotation:Number;
private var nextRotation:Number;

private
```

```
var increment:Number = 0.0625;
// private var increment:Number = 1;
private var dir:int = 1;

private var timer:Timer;

public function advanceState( node:Node ):Boolean
{
var fsmNode:FSMNode = node as FSMNode;
var advanceState:Boolean;
var proceed:Boolean = true;

if (!node)
return false;

switch (_state)
{
case INIT:
{
if (isNaN(crntRotation))
crntRotation = Math.random() * Math.PI * 2;
nextRotation = Math.random() * Math.PI * 2;
_state = SCANNING;
break;
}

case SCANNING:
{
var delta:Number = dir == 1 ? nextRotation - crntRotation : crntRotation - nextRotation;

if (delta <= increment * 2)
{
dir *= -1;
_state = PAUSED;

proceed = false;
break;
}

crntRotation += increment * dir;

break;
}

case PAUSED:
{
if (!timer)
{
timer = new Timer(Math.random() * 5000);
timer.addEventListener(TimerEvent.TIMER, onTimer);
timer.start();
```

```
}

break;
}
}

if (proceed)
rotateTurretGraphic(crntRotation);

return true;
}

private function onTimer( event:TimerEvent ):void
{
_state = INIT;

timer.removeEventListener(TimerEvent.TIMER, onTimer);
timer = null;
}

private function rotateTurretGraphic( rotation:Number, isRadians:Boolean = true ):void
{
if (!animation)
return;

var degrees:Number = isRadians ? (rotation / Math.PI) * 180 : rotation;
degrees = degrees % 360;

if (degrees < 0)
degrees += 360;

var num:int = degrees / 8.18 | 0;
animation.frame = num;

if (animation.render && animation.spritePack)
animation.render.updateFrame(animation.spritePack.getFrame(animation.label, num),
animation);
}

public function get component():IFSMComponent { return this; }

public function get state():int { return 0; }
public function set state( v:int ):void {}

public function destroy():void
{
animation = null;
_state = INIT;
}
}
```

```
}

-----------------------------------
File 184: igw\com\game\entity\components\shared\render\BarRender.as
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.BlendMode;
import flash.display.Sprite;

public class BarRender extends Sprite implements IRender
{
private var _back:Bitmap;
private var _fore:Bitmap;
private var _label:Label;
private var _scaleX:Number;
private var _string:String;

public function BarRender()
{
_back = new Bitmap();
_fore = new Bitmap();
_scaleX = 0;
addChild(_back);
addChild(_fore);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (animation.ready && !_back.bitmapData)
{
_back.bitmapData = BitmapData(animation.spritePack.getFrame(animation.label, 0));
_fore.bitmapData = BitmapData(animation.spritePack.getFrame(animation.label + "Green", 0));
_fore.scaleX = _scaleX;
}
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
scaleX = sx;
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5
```

```
):void {}
public function removeGlow():void {}

public function get color():uint { return 0; }
public function set color( value:uint ):void {}

override public function get scaleX():Number { return _fore.scaleX; }
override public function set scaleX( value:Number ):void
{
_scaleX = value;
_fore.scaleX = value;
}

override public function get scaleY():Number { return 1; }
override public function set scaleY( value:Number ):void {}

public function set text( v:String ):void
{
if (!_label)
{
_label = new Label(12, 0xecffff, 60, 22, true, 1);
_label.constrictTextToSize = false;
}
if (v && v != _string)
{
_label.text = v;
_string = v;
}
_label.x = 2;
_label.y = -2;
_fore.x = _fore.y = 3;
addChild(_label);
}

public function destroy():void
{
_scaleX = 0;
_back.bitmapData = null;
_fore.bitmapData = null;
_fore.x = _fore.y = 0;
if (_label)
{
_label.destroy();
removeChild(_label);
_label = null;
_string = null;
}
alpha = 1;
visible = true;
rotation = 0;
blendMode
```

```
= BlendMode.NORMAL;
}
}
}

-----------------------------------
File 185: igw\com\game\entity\components\shared\render\BarRenderStarling.as
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;

import org.starling.display.BlendMode;
import org.starling.display.Image;
import org.starling.display.Sprite;
import org.starling.text.TextField;
import org.starling.textures.Texture;

public class BarRenderStarling extends Sprite implements IRender
{
private var _back:Image;
private var _fore:Image;
private var _scaleX:Number;
private var _string:String;
private var _text:TextField;

public function BarRenderStarling()
{
_back = new Image(RenderStarling.DEFAULT_TEXTURE);
_fore = new Image(RenderStarling.DEFAULT_TEXTURE);
_scaleX = 0;
addChild(_back);
addChild(_fore);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (animation.ready)
{
_back.texture = Texture(animation.spritePack.getFrame(animation.label, 0));
_back.readjustSize();
_fore.texture = Texture(animation.spritePack.getFrame(animation.label + "Green", 0));
_fore.readjustSize();
_fore.scaleX = _scaleX;
}
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
scaleX
```

```
= sx;
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void {}
public function removeGlow():void {}

public function get color():uint { return 0; }
public function set color( value:uint ):void {}

override public function get scaleX():Number { return _fore.scaleX; }
override public function set scaleX( value:Number ):void
{
_scaleX = value;
_fore.scaleX = value;
}

public function set text( text:String ):void
{
if (!_text || text == null)
{
if (_text && contains(_text))
removeChild(_text);
_text = new TextField(66, 14, '', "Open Sans", 12, 0xecffff);
_text.x = 0;
_text.y = 1;
_fore.x = _fore.y = 3;
addChild(_text);
}
if (text != null && text != _string)
{
_string = text;
_text.text = text;
}
}

public function destroy():void
{
_scaleX = 0;
//texture = null;
alpha = 1;
visible = true;
rotation = 0;
if (_text)
{
if (contains(_text))
removeChild(_text);
_text = null;
}
_fore.x = _fore.y = 0;
_string
```

```
= null;
blendMode = BlendMode.NORMAL;
}
}
}

-----------------------------------
File 186: igw\com\game\entity\components\shared\render\ButtonRender.as
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.BlendMode;
import flash.display.Sprite;

public class ButtonRender extends Sprite implements IRender
{
private var _back:Bitmap;
private var _label:Label;
private var _scaleX:Number;

public function ButtonRender()
{
_back = new Bitmap();
_scaleX = 0;
addChild(_back);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (animation.ready && !_back.bitmapData)
{
_back.bitmapData = BitmapData(animation.spritePack.getFrame(animation.label, 0));
}
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
scaleX = sx;
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void {}
public function removeGlow():void {}

public
```

```
function get color():uint { return 0; }
public function set color( value:uint ):void {}

override public function get scaleY():Number { return 1; }
override public function set scaleY( value:Number ):void {}

public function set text( v:String ):void
{
if (!_label)
{
_label = new Label(12, 0xecffff, 60, 22, true, "Open Sans");
_label.constrictTextToSize = false;
}
_label.text = v;
_label.x = 2;
addChild(_label);
}

public function destroy():void
{
_scaleX = 0;
_back.bitmapData = null;
if (_label)
{
_label.destroy();
removeChild(_label);
_label = null;
}
alpha = 1;
visible = true;
rotation = 0;
blendMode = BlendMode.NORMAL;
}
}
}

-------------------------------------
File 187: igw\com\game\entity\components\shared\render\ButtonRenderStarling.as
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;

import org.starling.display.BlendMode;
import org.starling.display.Image;
import org.starling.display.Sprite;
import org.starling.text.TextField;
import org.starling.textures.Texture;

public class ButtonRenderStarling extends Sprite implements IRender
{
```

```
private var _back:Image;
private var _scaleX:Number;
private var _text:TextField;

public function ButtonRenderStarling()
{
_back = new Image(RenderStarling.DEFAULT_TEXTURE);
_scaleX = 0;
addChild(_back);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (animation.ready)
{
_back.texture = Texture(animation.spritePack.getFrame(animation.label, 0));
_back.readjustSize();
}
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
scaleX = sx;
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void {}
public function removeGlow():void {}

public function get color():uint { return 0; }
public function set color( value:uint ):void {}

public function set text( text:String ):void
{
if (!_text)
_text = new TextField(66, 14, '', "Open Sans", 12, 0xecffff);
else
{
removeChild(_text);
_text = new TextField(66, 14, '', "Open Sans", 12, 0xecffff);
}
_text.x = 0;
_text.y = 1;
_text.text = text;
addChild(_text);
}

public function destroy():void
{
```

```
_scaleX = 0;
//texture = null;
alpha = 1;
visible = true;
rotation = 0;
if (_text)
{
if (contains(_text))
removeChild(_text);
_text = null;
}
blendMode = BlendMode.NORMAL;
}
}
}
```

File 188: igw\com\game\entity\components\shared\render\NameRender.as

```
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.game.entity.factory.VCFactory;
import com.ui.core.component.label.Label;

import flash.display.BlendMode;
import flash.display.Sprite;

public class NameRender extends Sprite implements IRender
{
private var _label:Label;

public function NameRender()
{
_label = new Label(VCFactory.FONT_SIZE, VCFactory.FONT_COLOR,
VCFactory.TEXT_WIDTH, VCFactory.TEXT_HEIGHT, true, 1);
_label.bold = true;
_label.useLocalization = false;
_label.leading = -4;
_label.constrictTextToSize = false;
addChild(_label);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void {}

public
```

```
function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number = .5
):void {}
public function removeGlow():void {}

public function get color():uint { return (_label != null) ? _label.textColor : 0; }
public function set color( value:uint ):void { if (_label) _label.textColor = value; }

override public function get height():Number { return VCFactory.TEXT_HEIGHT; }
override public function get width():Number { return VCFactory.TEXT_WIDTH; }

public function set text( v:String ):void { _label.text = v; }

public function destroy():void
{
_label.text = '';
alpha = scaleX = scaleY = 1;
visible = true;
rotation = 0;
blendMode = BlendMode.NORMAL;
}
}
}
```

File 189: igw\com\game\entity\components\shared\render\NameRenderStarling.as

```
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.game.entity.factory.VCFactory;

import org.starling.display.BlendMode;
import org.starling.display.Sprite;
import org.starling.text.TextField;
import org.starling.utils.VAlign;

public class NameRenderStarling extends Sprite implements IRender
{
private var _label:TextField;
private var _text:String = '';
private var _color:uint;

public function NameRenderStarling()
{
_color = VCFactory.FONT_COLOR;
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{

}
```

```
public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{

}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void {}
public function removeGlow():void {}

public function get color():uint { return (_label != null) ? _label.color : _color; }
public function set color( value:uint ):void
{
if (_label)
_label.color = value;

_color = value;
}

override public function get height():Number { return VCFactory.TEXT_HEIGHT; }
override public function get width():Number { return VCFactory.TEXT_WIDTH; }

public function set text( v:String ):void
{
if (!_label)
{
removeChild(_label);
_label = new TextField(VCFactory.TEXT_WIDTH, VCFactory.TEXT_HEIGHT, '',
'OpenSansBoldBitmap', VCFactory.FONT_SIZE, _color);
_label.kerning = true;
_label.vAlign = VAlign.TOP;
addChild(_label);
}
if (_text != v)
{
_label.text = v;
_text = v;
}
}

public function destroy():void
{
alpha = scaleX = scaleY = 1;
visible = true;
rotation = 0;
blendMode = BlendMode.NORMAL;
}
}
}
```

File 190: igw\com\game\entity\components\shared\render\Render.as

```
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.BlendMode;
import flash.filters.ColorMatrixFilter;
import flash.filters.GlowFilter;
import flash.geom.Matrix;

public class Render extends Bitmap implements IRender
{
private var _colorFilter:ColorMatrixFilter;
private var _colorMatrix:Array;
private var _filters:Array;
private var _matrix:Matrix;

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (image)
bitmapData = BitmapData(image);
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
if (!_matrix)
_matrix = new Matrix();
_matrix.identity();
_matrix.translate(-offsetX, -offsetY);
if (scaleFirst)
{
_matrix.scale(sx, sy);
_matrix.rotate(rot);
} else
{
_matrix.rotate(rot);
_matrix.scale(sx, sy);
}
_matrix.translate(x + offsetX, y + offsetY);
transform.matrix = _matrix;
smoothing = true;
}

public
```

```
function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number = .5
):void
{
if (_colorFilter == null)
initColorMode();
_filters.push(new GlowFilter(color, 1, 6 / resolution, 6 / resolution, strength));
filters = _filters;
}
public function removeGlow():void
{
if (!_filters)
return;
_filters.pop();
filters = _filters;
}

public function get color():uint { return transform.colorTransform.color; }
public function set color( value:uint ):void
{
if (value != 0xffffff)
{
if (_colorFilter == null)
initColorMode();
_colorMatrix[0] = (((value >> 16) & 0xFF) / 0xFF);
_colorMatrix[6] = (((value >> 8) & 0xFF) / 0xFF);
_colorMatrix[12] = ((value & 0xFF) / 0xFF);
_colorFilter.matrix = _colorMatrix;

filters = _filters;
} else
filters.length = 0;
}

/**
* Setting up the necessary objects to perform coloring just once here to avoid
* creating new ones each time the color is changed. this helps with performance
* and garbage collection
*/
private function initColorMode():void
{
_colorFilter = new ColorMatrixFilter();
_colorMatrix = [1, 0, 0, 0, 0,
0, 1, 0, 0, 0,
0, 0, 1, 0, 0,
0, 0, 0, 1, 0];
_filters = [_colorFilter];
}

public function destroy():void
{
if
```

```
(_matrix)
{
_matrix.identity();
transform.matrix = _matrix;
}
_matrix = null;
bitmapData = null;
alpha = scaleX = scaleY = 1;
visible = true;
rotation = 0;
smoothing = false;
blendMode = BlendMode.NORMAL;
_colorFilter = null;
_colorMatrix = null;
if (_filters)
{
_filters.length = 0;
filters = _filters;
_filters = null;
}
}
}
}
```

File 191: igw\com\game\entity\components\shared\render\Render3D.as

```
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;
import com.ui.core.ViewStack3D;

import flash.display.BitmapData;
import flash.geom.Matrix;
import flash.geom.Point;

import org.away3d.containers.ObjectContainer3D;
import org.away3d.entities.Mesh;
import org.away3d.events.AssetEvent;
import org.away3d.loaders.Loader3D;
import org.away3d.materials.TextureMaterial;
import org.away3d.primitives.CubeGeometry;
import org.away3d.textures.BitmapTexture;
import org.starling.display.DisplayObject;

public class Render3D extends ObjectContainer3D implements IRender
{
private static var _cubeMaterial:TextureMaterial;

private var _cube1:Mesh;
private
```

```
var _loader:Loader3D;
private var _matrix:Matrix;

//solider ant model
[Embed(source="/../../../../assets/assets/Ship.3ds", mimeType="application/octet-stream")]
public static var ShipModel:Class;

public function Render3D()
{
//Create a material for the cubes
if (_cubeMaterial == null)
{
var cubeBmd:BitmapData = new BitmapData(128, 128, false, 0x0);
cubeBmd.perlinNoise(7, 7, 5, 12345, true, true, 7, true);
_cubeMaterial = new TextureMaterial(new BitmapTexture(cubeBmd));
_cubeMaterial.gloss = 20;
_cubeMaterial.ambientColor = 0x808080;
_cubeMaterial.ambient = 1;
_cubeMaterial.lightPicker = ViewStack3D.lightPicker;
}

// Build the cubes for view 1
var cG:CubeGeometry = new CubeGeometry(80, 80, 80);
_cube1 = new Mesh(cG, _cubeMaterial);

// Add the cubes to view 1
//addChild(_cube1);

_loader = new Loader3D();
_loader.scale(.8);
_loader.addEventListener(AssetEvent.ASSET_COMPLETE, onAssetComplete);
_loader.loadData(new ShipModel());
addChild(_loader);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
_cube1.x = 40;
_cube1.y = 40;
_cube1.z = 40;
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{

}

public function hitTest( localPoint:Point, forTouch:Boolean = false ):DisplayObject
{
return
```

```
null;
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void {}
public function removeGlow():void {}

private function onAssetComplete( e:AssetEvent ):void
{
if (e.asset.assetType == "mesh")
{
Mesh(e.asset).material = _cubeMaterial;
}
}

public function get alpha():Number { return 0; }
public function set alpha( v:Number ):void {}

public function get blendMode():String { return null; }
public function set blendMode( v:String ):void {}

public function get color():uint { return 0; }
public function set color( value:uint ):void {}

public function get height():Number { return 100; }
public function set height( v:Number ):void {}

public function get rotation():Number { return 0; }
public function set rotation( v:Number ):void
{
v += 90;
var _tempRotation:Number = Math.atan2(Math.sin(v), Math.cos(v));
_tempRotation = (_tempRotation / Math.PI) * 180;
_tempRotation = _tempRotation % 360;
if (_tempRotation < 0)
_tempRotation += 360;
_loader.rotateTo(0, v, 0);
}

public function get width():Number { return 100; }
public function set width( v:Number ):void {}

override public function set x( val:Number ):void { super.x = -val; }

override public function set y( val:Number ):void { super.y = -val; }



public function destroy():void
{
if
```

```
(_matrix)
{
_matrix.identity();
}
_matrix = null;
visible = false;
dispose();
}
}
}

------------------------------------
File 192: igw\com\game\entity\components\shared\render\RenderSprite.as
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.BlendMode;
import flash.display.Sprite;
import flash.filters.ColorMatrixFilter;
import flash.filters.GlowFilter;
import flash.geom.Matrix;

public class RenderSprite extends Sprite implements IRender
{
private var _bitmap:Bitmap;
private var _colorFilter:ColorMatrixFilter;
private var _colorMatrix:Array;
private var _filters:Array;
private var _matrix:Matrix;

public function RenderSprite()
{
_bitmap = new Bitmap();
addChild(_bitmap);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (image)
_bitmap.bitmapData = BitmapData(image);
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
if (!_matrix)
_matrix
```

```
= new Matrix();
_matrix.identity();
_matrix.translate(-offsetX, -offsetY);
if (scaleFirst)
{
_matrix.scale(sx, sy);
_matrix.rotate(rot);
} else
{
_matrix.rotate(rot);
_matrix.scale(sx, sy);
}
_matrix.translate(x + offsetX, y + offsetY);
transform.matrix = _matrix;
_bitmap.smoothing = true;
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void
{
if (_colorFilter == null)
initColorMode();
_filters.push(new GlowFilter(color, 1, 6 / resolution, 6 / resolution, strength));
filters = _filters;
}
public function removeGlow():void
{
if (!_filters)
return;
_filters.pop();
filters = _filters;
}

public function get color():uint { return transform.colorTransform.color; }
public function set color( value:uint ):void
{
if (value != 0xffffff)
{
if (_colorFilter == null)
initColorMode();
_colorMatrix[0] = (((value >> 16) & 0xFF) / 0xFF);
_colorMatrix[6] = (((value >> 8) & 0xFF) / 0xFF);
_colorMatrix[12] = ((value & 0xFF) / 0xFF);
_colorFilter.matrix = _colorMatrix;

filters = _filters;
} else
filters.length = 0;
}

/**
```

```
* Setting up the necessary objects to perform coloring just once here to avoid
* creating new ones each time the color is changed. this helps with performance
* and garbage collection
*/
private function initColorMode():void
{
_colorFilter = new ColorMatrixFilter();
_colorMatrix = [1, 0, 0, 0, 0,
0, 1, 0, 0, 0,
0, 0, 1, 0, 0,
0, 0, 0, 1, 0];
_filters = [_colorFilter];
}

override public function get height():Number { return _bitmap.height; }
override public function get width():Number { return _bitmap.width; }

public function destroy():void
{
if (_matrix)
{
_matrix.identity();
transform.matrix = _matrix;
}
_matrix = null;
_bitmap.bitmapData = null;
_bitmap.smoothing = false;
alpha = scaleX = scaleY = 1;
visible = true;
rotation = 0;
blendMode = BlendMode.NORMAL;
_colorFilter = null;
_colorMatrix = null;
if (_filters)
{
_filters.length = 0;
filters = _filters;
_filters = null;
}
}
}
}
```

File 193: igw\com\game\entity\components\shared\render\RenderSpriteStarling.as

```
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;

import
```

```
flash.display.BitmapData;
import flash.geom.Matrix;
import flash.geom.Point;

import org.starling.display.BlendMode;
import org.starling.display.DisplayObject;
import org.starling.display.Image;
import org.starling.display.Sprite;
import org.starling.filters.BlurFilter;
import org.starling.textures.Texture;

public class RenderSpriteStarling extends Sprite implements IRender
{
private var _i:int;
private var _image:Image;
private var _matrix:Matrix;
private var _resize:Boolean;

public function RenderSpriteStarling()
{
if (!RenderStarling.DEFAULT_TEXTURE)
RenderStarling.DEFAULT_TEXTURE = Texture.fromBitmapData(new BitmapData(1, 1, false,
0), false);
_resize = true;
_image = new Image(RenderStarling.DEFAULT_TEXTURE);
addChild(_image);
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (image)
{
_image.texture = Texture(image);
if (_resize || forceResize)
{
_image.readjustSize()
_resize = false;
visible = true;
}
} else
{
_resize = true;
_image.texture = Texture(RenderStarling.DEFAULT_TEXTURE);
_image.readjustSize();
}
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
if
```

```
(!_matrix)
_matrix = new Matrix();
_matrix.identity();
_matrix.translate(-offsetX, -offsetY);
if (scaleFirst)
{
_matrix.scale(sx, sy);
_matrix.rotate(rot);
} else
{
_matrix.rotate(rot);
_matrix.scale(sx, sy);
}
_matrix.translate(x + offsetX, y + offsetY);
transformationMatrix = _matrix;
}

override public function hitTest( localPoint:Point, forTouch:Boolean = false ):DisplayObject
{
localPoint = globalToLocal(localPoint)
return super.hitTest(localPoint, forTouch);
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void { filter = BlurFilter.createGlow(color, strength, blur, resolution); }
public function removeGlow():void { if (filter) filter.dispose(); filter = null; }

public function get color():uint { return _image.color; }
public function set color( value:uint ):void
{
_image.color = value;
for (_i = 0; _i < numChildren; _i++)
{
Image(getChildAt(_i)).color = value;
}
}

override public function get height():Number { return _image.height; }
override public function get width():Number { return _image.width; }

public function destroy():void
{
if (_matrix)
{
_matrix.identity();
transformationMatrix = _matrix;
}
blendMode = BlendMode.NORMAL;
_matrix = null;
_resize = true;
alpha
```

```
= scaleX = scaleY = 1;
visible = false;
rotation = 0;
color = 0xffffff;
dispose();
}
}
}
```

File 194: igw\com\game\entity\components\shared\render\RenderStarling.as

```
package com.game.entity.components.shared.render
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.IRender;

import flash.display.BitmapData;
import flash.geom.Matrix;
import flash.geom.Point;

import org.starling.display.BlendMode;
import org.starling.display.DisplayObject;
import org.starling.display.Image;
import org.starling.filters.BlurFilter;
import org.starling.textures.Texture;

public class RenderStarling extends Image implements IRender
{
public static var DEFAULT_TEXTURE:Texture;

private var _matrix:Matrix;
private var _resize:Boolean;

public function RenderStarling()
{
if (!DEFAULT_TEXTURE)
DEFAULT_TEXTURE = Texture.fromBitmapData(new BitmapData(1, 1, false, 0), false);
super(DEFAULT_TEXTURE);
_resize = true;
}

public function updateFrame( image:*, animation:Animation, forceResize:Boolean = false ):void
{
if (image)
{
texture = Texture(image);
if (_resize || forceResize)
{
readjustSize();
_resize = false;
visible
```

```
= true;
}
} else
{
_resize = true;
texture = Texture(DEFAULT_TEXTURE);
readjustSize();
}
}

public function applyTransform( rot:Number, sx:Number, sy:Number, scaleFirst:Boolean,
offsetX:Number, offsetY:Number ):void
{
if (!_matrix)
_matrix = new Matrix();
_matrix.identity();
_matrix.translate(-offsetX, -offsetY);
if (scaleFirst)
{
_matrix.scale(sx, sy);
_matrix.rotate(rot);
} else
{
_matrix.rotate(rot);
_matrix.scale(sx, sy);
}
_matrix.translate(x + offsetX, y + offsetY);
transformationMatrix = _matrix;
}

override public function hitTest( localPoint:Point, forTouch:Boolean = false ):DisplayObject
{
localPoint = globalToLocal(localPoint)
return super.hitTest(localPoint, forTouch);
}

public function addGlow( color:uint, strength:Number = 1, blur:Number = 1, resolution:Number =
.5 ):void { filter = BlurFilter.createGlow(color, strength, 1, resolution); }
public function removeGlow():void { if (filter) filter.dispose(); filter = null; }

public function destroy():void
{
if (_matrix)
{
_matrix.identity();
transformationMatrix = _matrix;
}
blendMode = BlendMode.NORMAL;
_matrix = null;
_resize = true;
alpha
```

```
= scaleX = scaleY = 1;
visible = false;
rotation = 0;
color = 0xffffff;
dispose();
}
}
}

-----------------------------------
File 195: igw\com\game\entity\components\shared\render\VisualComponent.as
package com.game.entity.components.shared.render
{
import org.ash.core.Entity;

public class VisualComponent
{
public var parent:Entity;

private var _children:Array = [];

public function init( parent:Entity ):void
{
this.parent = parent;
}

public function addChild( child:* ):void { _children.push(child); }

public function getChildAt( index:int ):*
{
if (index < _children.length)
return _children[index];
return null;
}

public function removeChildAt( index:int ):*
{
if (index < _children.length)
return _children.splice(index, 1)[0]
return null;
}

public function get numChildren():int { return _children.length; }

public function destroy():void
{
_children.length = 0;
parent = null;
}
}
}
```

File 196: igw\com\game\entity\components\starbase\Building.as
```
package com.game.entity.components.starbase
{
import com.model.starbase.BuildingVO;

public class Building
{
public var buildingVO:BuildingVO;
public var faction:String;

public function init( vo:BuildingVO ):void
{
buildingVO = vo;
faction = '';
}

public function destroy():void
{
buildingVO = null;
}
}
}
```

File 197: igw\com\game\entity\components\starbase\Construction.as
```
package com.game.entity.components.starbase
{
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.enum.TypeEnum;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.VCList;
import com.game.entity.components.shared.fsm.IFSMComponent;
import com.game.entity.nodes.shared.FSMNode;

import org.ash.core.Entity;
import org.ash.core.Node;

public class Construction implements IFSMComponent
{
public static const BEGIN:int = 0;
public static const BEAM_DESCEND:int = 1;
public static const STABLE:int = 2;
public static const BEAM_ASCEND:int = 3;
public static const END:int = 4;
public static const DONE:int = 5;
public static const RESTABILIZE:int = 6;

public
```

```
var alpha:Number;
public var beam:Entity;
public var beam2:Entity;
public var beamScaleY:Number;
public var glow:Entity;
public var mothership:Entity;
public var owner:Entity;
public var positionOffset:Number;
public var ypos:Number;

private var _alphaCount:Number;
private var _positionCount:Number;
private var _positionMod:Number;
private var _state:int;

public function Construction()
{
_alphaCount = 0;
_positionCount = positionOffset = 0;
_positionMod = 2;
_state = BEGIN;
}

public function advanceState( node:Node ):Boolean
{
var animation:Animation;
var cNode:FSMNode = FSMNode(node);
switch (_state)
{
case BEGIN:
adjustAlpha(0);
cNode.animation.alpha += .05;
if (cNode.animation.alpha >= 1)
{
cNode.animation.alpha = 1;
state = Construction.BEAM_DESCEND;
}
break;
case BEAM_DESCEND:
adjustAlpha(_alphaCount + 2);
animation = beam.get(Animation);
animation.scaleY += .05;
animation.render.scaleY += .05;
animation = beam2.get(Animation);
animation.scaleY += .05;
animation.render.scaleY += .05;
animation.alpha = 0;
animation.render.alpha = 0;
if (_alphaCount >= 100 && animation.render.scaleY >= beamScaleY)
state = Construction.STABLE;
if
```

```
(animation.render.scaleY >= beamScaleY)
{
animation.render.scaleY = beamScaleY;
animation = beam.get(Animation);
animation.render.scaleY = beamScaleY;
}
SoundController.instance.playSound(AudioEnum.AFX_BLD_CONSTRUCTION_START, 0.6, 0,
0);
break;
case STABLE:
adjustAlpha();
break;
case RESTABILIZE:
animation = beam.get(Animation);
animation.render.scaleY = beamScaleY;
animation = beam2.get(Animation);
animation.render.scaleY = beamScaleY;
cNode.animation.alpha = 1;
state = Construction.STABLE;
break;
case BEAM_ASCEND:
animation = beam.get(Animation);
animation.alpha = (animation.alpha > 0) ? animation.alpha - .1 : 0;
animation.render.alpha = animation.alpha;
var animation2:Animation = beam2.get(Animation);
animation2.alpha = (animation2.alpha > 0) ? animation2.alpha - .1 : 0;
animation2.render.alpha = animation2.alpha;
var animation3:Animation = glow.get(Animation);
animation3.alpha = (animation3.alpha > 0) ? animation3.alpha - .1 : 0;
animation3.render.alpha = animation3.alpha;
if (animation.alpha <= 0 && animation2.alpha <= 0 && animation3.alpha <= 0)
state = Construction.END;
SoundController.instance.playSound(AudioEnum.AFX_BLD_CONSTRUCTION_COMPLETE,
1.0, 0, 0);
break;
case END:
cNode.animation.alpha -= .1;
if (cNode.animation.alpha <= 0)
{
state = Construction.DONE;
var vcList:VCList = owner.get(VCList);
vcList.removeComponentType(TypeEnum.BUILDING_CONSTRUCTION);
return false;
}
break;
}
//adjust position
if (_state != DONE)
adjustPosition();

return
```

```
true;
}

private function adjustAlpha( value:Number = -1 ):void
{
if (value == -1)
_alphaCount += 1.5;
else
_alphaCount = value;
alpha = Math.abs(((_alphaCount / 100 | 0) % 2) - (_alphaCount % 100) * .01);

//adjust the alphas
var animation:Animation = beam.get(Animation);
animation.alpha = alpha;
animation.render.alpha = alpha;
animation = beam2.get(Animation);
animation.alpha = (value != -1) ? alpha : 1 - alpha;
animation.render.alpha = alpha;
animation = glow.get(Animation);
animation.alpha = alpha;
animation.render.alpha = alpha;
}

private function adjustPosition():void
{
_positionCount += .1;
positionOffset = ypos + _positionMod * Math.sin(_positionCount);

//adjust the positions
var animation:Animation = glow.get(Animation);
animation.render.y = positionOffset;
animation = mothership.get(Animation);
animation.render.y = positionOffset;
}

public function get component():IFSMComponent { return this; }

public function get state():int { return _state; }
public function set state( v:int ):void
{
_state = v;
switch (_state)
{
case BEAM_ASCEND:
_alphaCount % 100;
break;
}
}

public function destroy():void
{
```

```
alpha = _alphaCount = 0;
beam = null;
beam2 = null;
glow = null;
mothership = null;
owner = null;
_positionCount = positionOffset = 0;
state = BEGIN;
}
}
}
```

File 198: igw\com\game\entity\components\starbase\Platform.as

```
package com.game.entity.components.starbase
{
import com.model.starbase.BuildingVO;

public class Platform
{
public var buildingVO:BuildingVO;

public function init( vo:BuildingVO ):void
{
buildingVO = vo;
}

public function destroy():void
{
buildingVO = null;
}
}
}
```

File 199: igw\com\game\entity\components\starbase\State.as

```
package com.game.entity.components.starbase
{
import com.event.TransactionEvent;
import com.model.transaction.TransactionVO;

import org.ash.core.Entity;

public class State
{
public var component:Entity;
public var text:String;
private var _transactions:Vector.<TransactionVO>;

public
```

```
function get percentageDone():Number { return _transactions[0].percentComplete; }
public function get startTime():uint { return _transactions[0].began; }
public function get remainingTime():uint { return _transactions[0].timeRemainingMS; }
public function get endTime():uint { return _transactions[0].began + _transactions[0].timeMS; }

public function addTransaction( transaction:TransactionVO ):void
{
if (_transactions == null)
_transactions = new Vector.<TransactionVO>;

if (!alreadyAdded(transaction))
_transactions.push(transaction);

if (_transactions.length > 1)
_transactions.sort(orderItems);
}

private function alreadyAdded( transaction:TransactionVO ):Boolean
{
var alreadyAdded:Boolean;
var len:uint = _transactions.length;
for (var i:uint = 0; i < len; ++i)
{
if (_transactions[i].id == transaction.id && _transactions[i].type == transaction.type)
{
alreadyAdded = true;
break;
}
}

return alreadyAdded;
}

public function removeTransaction( transactionID:String ):void
{
var len:uint = _transactions.length;
for (var i:uint = 0; i < len; ++i)
{
if (_transactions[i].id == transactionID)
{
_transactions.splice(i, 1);
break;
}
}
if (_transactions.length > 1)
_transactions.sort(orderItems);
}

private function orderItems( transactionOne:TransactionVO, transactionTwo:TransactionVO
):Number
{
```

```
if (!transactionOne)
return -1;
if (!transactionTwo)
return 1;

var timeRemainingOne:uint = transactionOne.timeRemainingMS;
var timeRemainingTwo:uint = transactionTwo.timeRemainingMS;

if (timeRemainingOne < timeRemainingTwo)
return -1;
if (timeRemainingOne > timeRemainingTwo)
return 1;

return 0;
}

public function get showConstruction():Boolean
{
var transaction:TransactionVO;
for (var i:int = 0; i < _transactions.length; i++)
{
transaction = _transactions[i];
if (transaction.type == TransactionEvent.STARBASE_BUILDING_BUILD
|| transaction.type == TransactionEvent.STARBASE_BUILDING_UPGRADE
|| transaction.type == TransactionEvent.STARBASE_REFIT_BUILDING
|| transaction.type == TransactionEvent.STARBASE_REPAIR_BASE)
return true;
}
return false;
}

public function get transaction():TransactionVO { return _transactions[0]; }
public function get transactionCount():uint { return _transactions.length; }
public function get type():String { return _transactions[0].type; }

public function destroy():void
{
component = null;
text = null;
_transactions.length = 0;
_transactions = null;

}
}
}

------------------------------------
File 200: igw\com\game\entity\factory\AttackFactory.as
package com.game.entity.factory
{
```

```
import com.enum.CategoryEnum;
import com.enum.EntityMoveEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.ActiveDefense;
import com.game.entity.components.battle.Area;
import com.game.entity.components.battle.Beam;
import com.game.entity.components.battle.Drone;
import com.game.entity.components.battle.Modules;
import com.game.entity.components.battle.TrailFX;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;
import com.model.asset.AssetVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.sector.SectorModel;
import com.service.server.incoming.data.ActiveDefenseHitData;
import com.service.server.incoming.data.AreaAttackData;
import com.service.server.incoming.data.BeamAttackData;
import com.service.server.incoming.data.DroneAttackData;
import com.service.server.incoming.data.ProjectileAttackData;
import com.util.BattleUtils;

import flash.geom.Point;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public class AttackFactory extends BaseFactory implements IAttackFactory
{
private var _sectorModel:SectorModel;

public function createProjectile( owner:Entity, data:ProjectileAttackData ):Entity
{
//Determine the firing ship or building faction
var faction:String = FactionEnum.IGA;
if (owner)
{
var ownerDetail:Detail = owner.get(Detail);
if (ownerDetail.category == 'Building')
faction = _sectorModel.sectorFaction;
else
faction = ownerDetail.prototypeVO.getValue('faction');
} else
faction = (data.playerOwnerId == CurrentUser.id) ? CurrentUser.faction :
_sectorModel.sectorFaction;
```

```
//get the prototype and asset vo
var prototypeVO:IPrototype = _prototypeModel.getWeaponPrototype(data.weaponPrototype);
var assetVO:AssetVO;

if (prototypeVO.asset == 'MissilePod')
{
switch (faction)
{
case FactionEnum.IMPERIUM:
case FactionEnum.IGA:
assetVO = _assetModel.getEntityData(TypeEnum.IGA_MISSILE);
break;
case FactionEnum.SOVEREIGNTY:
assetVO = _assetModel.getEntityData(TypeEnum.SOV_MISSILE);
break;
case FactionEnum.TYRANNAR:
assetVO = _assetModel.getEntityData(TypeEnum.TYR_MISSILE);
break;
}
} else
assetVO = _assetModel.getEntityData(prototypeVO.asset);

var projectile:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.ATTACK, assetVO, prototypeVO);
projectile.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
var rot:Number = BattleUtils.instance.isoCrunchAngle(data.rotation);
pos.init(data.start.x, data.start.y, rot, LayerEnum.ATTACK);
projectile.add(pos);
//move component
var move:Move = ObjectPool.get(Move);
var fadeTime:Number = (data.fadeTime < 0) ? 3 : data.fadeTime;
if (data.guided)
{
move.init(30, EntityMoveEnum.LERPING);
move.fadeOut = data.finishTick - fadeTime;
} else
{
move.init(30, EntityMoveEnum.POINT_TO_POINT);
move.setPointToPoint(data.end.x, data.end.y, data.startTick, data.finishTick);
move.fadeOut = move.endTick - fadeTime;
}
projectile.add(move);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type,
```

```
assetVO.spriteName, true);
anim.scaleX = anim.scaleY = assetVO.scale;
anim.allowTransform = true;
projectile.add(anim);
//grid component
projectile.add(ObjectPool.get(Grid));
//assign the name
projectile.id = data.attackId;
//add missile trail component if this is a guided projectile and it is set to have a trail
if (detail.prototypeVO.getValue("trailLength") > 1)
{
// the missile trail component
var trail:TrailFX = ObjectPool.get(TrailFX);
trail.type = TypeEnum.MISSILE_TRAIL;
trail.maxSegments = prototypeVO.getValue("trailLength");
trail.color = parseInt(prototypeVO.getValue("trailColor"), 16);
trail.thickness = prototypeVO.getValue("trailThickness");
trail.lastPosition.setTo(pos.x, pos.y);
trail.alphaChange = 1 / trail.maxSegments;
projectile.add(trail);
}
//add to game
addEntity(projectile);
//Play audio
playSound(assetVO);
return projectile;
}

public function createBeam( data:BeamAttackData ):Entity
{
//get the prototype and asset vo
var prototypeVO:IPrototype = _prototypeModel.getWeaponPrototype(data.weaponPrototype);
var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);
var beam:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.ATTACK, assetVO, prototypeVO, data.entityOwnerId);
beam.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.start.x, data.start.y, 0, LayerEnum.ATTACK);
beam.add(pos);
//beam component
var beamC:Beam = ObjectPool.get(Beam);
beamC.init(data.entityOwnerId, data.targetEntityId, data.sourceAttachPoint,
data.targetAttachPoint, data.targetScatterX, data.targetScatterY, data.maxRange,
data.attackHit, .2);
beamC.baseWidth = assetVO.type ==
TypeEnum.TURRET_BEAM_POINT_DEFENSE_CLUSTER ? 128 : 256;
beamC.hitLocationX = data.hitLocation.x;
beamC.hitLocationY
```

```
= data.hitLocation.y;
beamC.hitTarget = data.hitTarget;
beam.add(beamC);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName);
anim.scaleX = 0;
anim.scaleY = assetVO.scale;
anim.allowTransform = true;
beam.add(anim);
//grid component
beam.add(ObjectPool.get(Grid));
//assign the name
beam.id = data.attackId;
//add to game
addEntity(beam);
//Play SoundFX
playSound(assetVO);
return beam;
}

public function createDrone( owner:Entity, data:DroneAttackData ):Entity
{
//Determine the firing ship or building faction
var ownerDetail:Detail = owner.get(Detail);
var faction:String = '';

if (ownerDetail.category == 'Building')
faction = _sectorModel.sectorFaction;
else
faction = ownerDetail.prototypeVO.getValue('faction');

//get the prototype and asset vo
var prototypeVO:IPrototype = _prototypeModel.getWeaponPrototype(data.weaponPrototype);
var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);
var attack:Entity = createEntity();

//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.ATTACK, assetVO, prototypeVO);
attack.add(detail);

//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.start.x, data.start.y, data.rotation * 0.0174532925, LayerEnum.ATTACK);
attack.add(pos);

//move component
var move:Move = ObjectPool.get(Move);
move.init(30, EntityMoveEnum.LERPING);
attack.add(move);
```

```
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, true);
anim.allowTransform = true;
anim.scaleX = anim.scaleY = assetVO.scale;
attack.add(anim);

//drone component
var drone:Drone = ObjectPool.get(Drone);
drone.init(data.entityOwnerId, data.targetEntityId);
drone.currentTick = 0;
drone.fireDuration = prototypeVO.getValue("fireDuration");
drone.minWeaponTime = prototypeVO.getValue("minWeaponTime") * 30.0;
drone.maxWeaponTime = prototypeVO.getValue("maxWeaponTime") * 30.0;
drone.weaponProto = prototypeVO.getValue("weaponProto");
attack.add(drone);

//grid component
attack.add(ObjectPool.get(Grid));

//assign the name
attack.id = data.attackId;

//add to game
addEntity(attack);

//Play SoundFX
playSound(assetVO);
return attack;
}

public function createArea( shipID:String, data:AreaAttackData ):Entity
{
//get the prototype and asset vo
var prototypeVO:IPrototype = _prototypeModel.getWeaponPrototype(data.weaponPrototype);
var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);
var ship:Entity = _game.getEntity(shipID);
var area:Entity = createEntity();

//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.ATTACK, assetVO, prototypeVO, shipID);
area.add(detail);

//area component
var areaC:Area = ObjectPool.get(Area);
areaC.init(shipID, data.sourceAttachPoint, 1500, prototypeVO);
area.add(areaC);

//position
```

```
component
var pos:Position = ObjectPool.get(Position);
pos.init(data.start.x, data.start.y, 0, LayerEnum.ATTACK);
if (areaC.rotateWithSource && ship)
pos.rotation = BattleUtils.instance.getAttachPointRotation(ship, areaC.sourceAttachPoint)
var randomRot:Boolean = prototypeVO.getValue("randomRot");
if (randomRot)
pos.rotation += (Math.random() * 6.28318531);
area.add(pos);

//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, !areaC.useBeamDynamics);
anim.scaleX = areaC.startScaleX;
anim.scaleY = areaC.startScaleY;
anim.allowTransform = true;
var randomAnim:Boolean = prototypeVO.getValue("randomAnim");
if (randomAnim)
anim.randomStart = true;
var color:String = prototypeVO.getValue("color");
if (color)
anim.color = parseInt(color, 16);
var stretch:Boolean = prototypeVO.getValue("stretchFrames");
if (stretch)
anim.duration = (areaC.animLength > areaC.duration) ? areaC.animLength : areaC.duration;
area.add(anim);

//animation component
area.add(ObjectPool.get(Grid));

//assign the name
area.id = data.attackId;

//add to game
addEntity(area);
//Play SoundFX
playSound(assetVO);
return area;
}

public function createActiveDefenseInterceptor( owner:Entity, attachPoint:String, x:int, y:int
):Entity
{
//get the prototype and asset vo
var modules:Modules = Modules(owner.get(Modules));
var prototypeVO:IPrototype = modules.getModuleByAttachPoint(attachPoint);
if (prototypeVO == null)
return null;
var activeDefenseType:int = ActiveDefense.BEAM;
if (prototypeVO.itemClass == TypeEnum.DEFLECTOR_SCREEN)
activeDefenseType
```

```
= ActiveDefense.FLAK;
else if (prototypeVO.itemClass == TypeEnum.DISTORTION_WEB)
activeDefenseType = ActiveDefense.SHIELD;
if (activeDefenseType == ActiveDefense.SHIELD)
return null;
var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);
var interceptor:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.ATTACK, assetVO, prototypeVO, owner.id);
interceptor.add(detail);
//activeDefense component
var activeDefense:ActiveDefense = ObjectPool.get(ActiveDefense);
activeDefense.init(activeDefenseType, owner, attachPoint);
activeDefense.hitLocationX = x;
activeDefense.hitLocationY = y;
interceptor.add(activeDefense);
//position component
var sourceLoc:Point = new Point(x, y);
if (activeDefenseType == ActiveDefense.BEAM)
BattleUtils.instance.getAttachPointLocation(owner, attachPoint, sourceLoc);
var pos:Position = ObjectPool.get(Position);
pos.init(sourceLoc.x, sourceLoc.y, 0, LayerEnum.ATTACK);
interceptor.add(pos);
//animation component
var anim:Animation = ObjectPool.get(Animation);
var center:Boolean = (activeDefenseType == ActiveDefense.FLAK) ? true : false;
anim.init(assetVO.type, assetVO.spriteName, center, 0, 30, true);
anim.scaleX = 0;
anim.scaleY = assetVO.scale;
anim.allowTransform = true;
interceptor.add(anim);
//assign the name
interceptor.id = "AD" + owner.id + attachPoint + x + y;
//add to game
addEntity(interceptor);
//Play SoundFX
playSound(assetVO);
return interceptor;
}

public function cleanAttack( attack:Entity ):void
{
if (attack.has(Beam))
ObjectPool.give(attack.remove(Beam));
if (attack.has(Move))
ObjectPool.give(attack.remove(Move));
if (attack.has(Area))
ObjectPool.give(attack.remove(Area));
if (attack.has(Drone))
ObjectPool.give(attack.remove(Drone));
```

```
if (attack.has(TrailFX))
ObjectPool.give(attack.remove(TrailFX));
ObjectPool.give(attack.remove(Grid));
}

public function destroyAttack( attack:Entity ):void
{
destroyEntity(attack);
if (attack.has(Beam))
ObjectPool.give(attack.remove(Beam));
if (attack.has(Move))
ObjectPool.give(attack.remove(Move));
if (attack.has(ActiveDefense))
ObjectPool.give(attack.remove(ActiveDefense));
if (attack.has(Area))
ObjectPool.give(attack.remove(Area));
if (attack.has(Drone))
ObjectPool.give(attack.remove(Drone));
if (attack.has(TrailFX))
ObjectPool.give(attack.remove(TrailFX));
if (attack.has(Grid))
ObjectPool.give(attack.remove(Grid));
ObjectPool.give(attack.remove(Detail));
ObjectPool.give(attack.remove(Position));
ObjectPool.give(attack.remove(Animation));
}

[Inject]
public function set sectorModel( value:SectorModel ):void
{
_sectorModel = value;
}

}
}

------------------------------------
File 201: igw\com\game\entity\factory\BackgroundFactory.as
package com.game.entity.factory
{
import com.enum.CategoryEnum;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.systems.shared.background.BackgroundItem;
import com.model.asset.AssetVO;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public
```

```
class BackgroundFactory extends BaseFactory implements IBackgroundFactory
{
public function createBackground( item:BackgroundItem ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(item.type);
var backgroundItem:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.BACKGROUND, assetVO);
backgroundItem.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(item.x, item.y, 0, item.layer, item.parallaxSpeed);
backgroundItem.add(pos);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(item.type, item.label, false, 0, 30, true);
anim.scaleX = anim.scaleY = item.scale;
anim.allowTransform = true;
backgroundItem.add(anim);
//assign the name
backgroundItem.id = item.id;
//add to the game
addEntity(backgroundItem);
return backgroundItem;
}

public function destroyBackground( backgroundItem:Entity ):void
{
if (backgroundItem == null)
return;
destroyEntity(backgroundItem);
ObjectPool.give(backgroundItem.remove(Detail));
ObjectPool.give(backgroundItem.remove(Position));
ObjectPool.give(backgroundItem.remove(Animation));
}
}
}
```

File 202: igw\com\game\entity\factory\BaseFactory.as

```
package com.game.entity.factory
{
import com.controller.sound.SoundController;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.PrototypeModel;

import flash.events.IEventDispatcher;

import
```

```
org.ash.core.Entity;
import org.ash.core.Game;
import org.shared.ObjectPool;
import org.swiftsuspenders.Injector;

public class BaseFactory
{
protected var _assetModel:AssetModel;
protected var _eventDispatcher:IEventDispatcher;
protected var _game:Game;
protected var _injector:Injector;
protected var _prototypeModel:PrototypeModel;
protected var _soundController:SoundController;

protected function createEntity():Entity
{
return ObjectPool.get(Entity);
}

protected function addEntity( entity:Entity ):void
{
_game.addEntity(entity);
}

protected function playSound( assetVO:AssetVO ):void
{
if (assetVO && assetVO.audio)
_soundController.playSound(assetVO.audio, assetVO.volume, 0, assetVO.loops);
}

public function destroyEntity( entity:Entity ):void
{
_game.removeEntity(entity);
ObjectPool.give(entity);
}

[Inject]
public function set assetModel( value:AssetModel ):void { _assetModel = value; }
[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
[Inject]
public function set game( value:Game ):void { _game = value; }
[Inject]
public function set inject( value:Injector ):void { _injector = value; }
[Inject]
public function set prototypeModel( value:PrototypeModel ):void { _prototypeModel = value; }
[Inject]
public function set soundController( value:SoundController ):void { _soundController = value; }
}
}

-----------------------------------
```

File 203: igw\com\game\entity\factory\IAttackFactory.as

```
package com.game.entity.factory
{
import com.service.server.incoming.data.AreaAttackData;
import com.service.server.incoming.data.BeamAttackData;
import com.service.server.incoming.data.DroneAttackData;
import com.service.server.incoming.data.ProjectileAttackData;

import org.ash.core.Entity;

public interface IAttackFactory
{
function createProjectile( ship:Entity, data:ProjectileAttackData ):Entity;

function createBeam( data:BeamAttackData ):Entity;

function createDrone( ship:Entity, data:DroneAttackData ):Entity;

function createActiveDefenseInterceptor( owner:Entity, attachPoint:String, x:int, y:int ):Entity;

function cleanAttack( attack:Entity ):void;

function destroyAttack( attack:Entity ):void;

function createArea( shipID:String, data:AreaAttackData ):Entity;
}
}
```

File 204: igw\com\game\entity\factory\IBackgroundFactory.as

```
package com.game.entity.factory
{
import com.game.entity.systems.shared.background.BackgroundItem;

import org.ash.core.Entity;

public interface IBackgroundFactory
{
function createBackground( item:BackgroundItem ):Entity;

function destroyBackground( backgroundItem:Entity ):void
}
}
```

File 205: igw\com\game\entity\factory\IInteractFactory.as

```
package com.game.entity.factory
{
import flash.geom.Point;
import
```

```
flash.geom.Rectangle;

import org.ash.core.Entity;

public interface IInteractFactory
{
function showSelection( target:Entity, selector:Entity, x:int = 0, y:int = 0, inBattle:Boolean =
false):Entity;

function showMultiShipSelection( target:Entity, selectionRect:Rectangle ):Entity;

function createRouteLine( entity:Entity, destination:Point ):Entity;

function createRange( entity:Entity ):Entity;

function createShipRange( entity:Entity ):Entity;

function clearRanges():void;

function destroyInteractEntity( entity:Entity ):Entity;
}
}

-----------------------------------
File 206: igw\com\game\entity\factory\InteractFactory.as
package com.game.entity.factory
{
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.components.starbase.Building;
import com.model.asset.AssetVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.model.starbase.BuildingVO;
import com.util.AllegianceUtil;
import com.util.RangeBuilder;
import com.util.RouteLineBuilder;

import flash.geom.Point;
import
```

```
flash.geom.Rectangle;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public class InteractFactory extends BaseFactory implements IInteractFactory
{
private var _id:int = 1;
private var _label:String;
private var _rangeBuilder:RangeBuilder;
private var _routeLineBuilder:RouteLineBuilder;
private var _playerModel:PlayerModel;
private var _sectorModel:SectorModel;
private var _type:String;

public function showSelection( target:Entity, selector:Entity, x:int = 0, y:int = 0 , inBattle:Boolean
= false):Entity
{
var animation:Animation;
var color:uint;
var isNew:Boolean = selector == null
var position:Position;

var faction:String = CurrentUser.faction;
if(inBattle)
faction = CurrentUser.battleFaction;

switch (faction)
{
case FactionEnum.IGA:
color = 0x3FD2FF;
break;
case FactionEnum.IMPERIUM:
color = 0x00ff00;
break;
case FactionEnum.SOVEREIGNTY:
color = 0xAF4DFF;
break;
case FactionEnum.TYRANNAR:
color = 0xFFAC3F;
break;
}

//if we don't have a target than we assume we're placing a destination
if (!target)
{
if (isNew)
selector = createInteractEntity("Destinationicon", TypeEnum.SHIELD);
//position component
if (isNew)
{
```

```
position = ObjectPool.get(Position);
position.init(x, y, 0, LayerEnum.VFX);
selector.add(position);
} else
{
position = selector.get(Position);
position.x = x;
position.y = y;
}
animation = selector.get(Animation);
animation.scaleX = animation.scaleY = 0.75;
animation.color = color;
//add to game
if (isNew)
_game.addEntity(selector);
return selector;
}

var detail:Detail = target.get(Detail);
if (detail.category == CategoryEnum.BUILDING && detail.type == TypeEnum.FORCEFIELD)
return null;
//determine which sprite to show
if (detail.category == CategoryEnum.SHIP || detail.category == CategoryEnum.BUILDING)
{
_label = target.has(Owned) ? "Selector" : "EnemySelector";
_type = TypeEnum.SHIELD;
color = 0xffffff;
} else
{
_label = "BaseSelector";
_type = TypeEnum.SHIELD;
}
if (selector && Animation(selector.get(Animation)).label != _label)
{
destroyInteractEntity(selector);
selector = null;
isNew = true;
}
if (isNew)
selector = createInteractEntity(_label, _type);

//update scale and color
animation = selector.get(Animation);
animation.color = color;
if (detail.category == CategoryEnum.SHIP)
{
if (target.has(Owned))
animation.color = AllegianceUtil.instance.getFactionColor(faction);
animation.scaleX = animation.scaleY = detail.assetVO.radius / 60;
if
```

```
(animation.scaleX < 1)
animation.scaleX = animation.scaleY = 1;
} else if (detail.category == CategoryEnum.BUILDING)
{
animation.scaleX = animation.scaleY = Building(target.get(Building)).buildingVO.sizeX / 5;
if (animation.scaleX < 1.2)
animation.scaleX = animation.scaleY = 1.2;
} else
{
switch (detail.type)
{
case TypeEnum.DERELICT_IGA:
case TypeEnum.DERELICT_SOVEREIGNTY:
case TypeEnum.DERELICT_TYRANNAR:
animation.scaleX = animation.scaleY = .5;
break;
default:
animation.scaleX = animation.scaleY = 1;
break;
}
}

//update or add the position component
if (isNew)
{
var oldPos:Position = target.get(Position);
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.MISC;
newPos.rotation = 0;
newPos.ignoreRotation = true;
oldPos.addLink(newPos);
selector.add(newPos);
} else
{
position = target.get(Position);
var selectorPos:Position = selector.get(Position);
selectorPos.linkedTo.removeLink(selectorPos);
selectorPos.x = position.x;
selectorPos.y = position.y;
selectorPos.dirty = true;
position.addLink(selectorPos);
}

//add to game
if (isNew)
_game.addEntity(selector);
return selector;
}

public function showMultiShipSelection( target:Entity, selectionRect:Rectangle ):Entity
{
```

```
var name:String = _rangeBuilder.drawShipSelectionBox();
var position:Position;

if (!target)
{
target = createInteractEntity(name, TypeEnum.SHIP_SELECTION_RANGE);
position = ObjectPool.get(Position);
position.init(selectionRect.x, selectionRect.y, 0, LayerEnum.RANGE);
target.add(position);
_game.addEntity(target);
} else
{
position = target.get(Position);
position.x = selectionRect.x;
position.y = selectionRect.y;
position.dirty = true;
}

var animation:Animation = target.get(Animation);
animation.scaleX = selectionRect.width / RangeBuilder.SELECTION_SIZE;
animation.scaleY = selectionRect.height / RangeBuilder.SELECTION_SIZE;

position.x += selectionRect.width / 2;
position.y += selectionRect.height / 2;

return target;
}

public function createRouteLine( ship:Entity, destination:Point ):Entity
{
_routeLineBuilder.drawRouteLine();
var routeLine:Entity = createInteractEntity(TypeEnum.ROUTE_LINE, TypeEnum.ROUTE_LINE);
var position:Position = (ship.get(Position) as Position).clone();
position.layer = LayerEnum.RANGE;
position.clearRotation();
routeLine.add(position);

var animation:Animation = routeLine.get(Animation);
animation.allowTransform = true;
animation.center = false;
animation.offsetY = RouteLineBuilder.OUTLINE_WIDTH / 2;

RouteLineBuilder.adjustRotation(routeLine, destination);
RouteLineBuilder.updateRouteLine(routeLine, ship);

_game.addEntity(routeLine);
return routeLine;
}

public
```

```
function createRange( entity:Entity ):Entity
{
var name:String;
var buildingVO:BuildingVO = Building(entity.get(Building)).buildingVO;
var scale:Number = 1;
if (buildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM)
{
var slots:Array = buildingVO.getValue("slots");
if (slots && buildingVO.modules && buildingVO.modules[slots[0]] != null)
{
var proto:IPrototype =
PrototypeModel.instance.getWeaponPrototype(buildingVO.modules[slots[0]].name);
var maxRange:Number = proto.getValue('maxRange');
var minRange:Number = proto.getValue('minRange');
scale = (maxRange > 1000) ? maxRange / 1000 : 1;
maxRange = maxRange / scale;
minRange = minRange / scale;
name = _rangeBuilder.drawStarbaseRange(maxRange, minRange);
} else
return null;
} else if (buildingVO.itemClass == TypeEnum.PYLON)
name = _rangeBuilder.drawPylonRange(buildingVO.shieldRadius);
else
name = _rangeBuilder.drawStarbaseRange(buildingVO.shieldRadius);

var range:Entity = createInteractEntity(name, TypeEnum.BASE_RANGE);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.RANGE;
oldPos.addLink(newPos);
range.add(newPos);
//animation component
var anim:Animation = range.get(Animation);
anim.allowTransform = true;
anim.transformScaleFirst = false;
anim.center = false;
anim.scaleX = scale;
anim.scaleY = .5 * scale;
var center:Point = _rangeBuilder.getCenter(name);
anim.offsetX = center.x;
anim.offsetY = center.y;
range.add(anim);
//grid component
range.add(ObjectPool.get(Grid));
//assign the name
range.id = id;
//add to game
_game.addEntity(range);
return range;
}
```

```
public function createShipRange( entity:Entity ):Entity
{
var name:String = Ship(entity.get(Ship)).rangeReference;
var range:Entity = createInteractEntity(name, TypeEnum.SHIP_RANGE);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.RANGE;
oldPos.addLink(newPos);
range.add(newPos);
//animation component
var anim:Animation = range.get(Animation);
anim.allowTransform = true;
anim.transformScaleFirst = false;
anim.center = false;
anim.scaleY = .5;
var center:Point = _rangeBuilder.getCenter(name);
anim.offsetX = center.x;
anim.offsetY = center.y;
range.add(anim);
//grid component
range.add(ObjectPool.get(Grid));
//add to game
_game.addEntity(range);
return range;
}

public function clearRanges():void
{
_rangeBuilder.cleanup();
}

private function createInteractEntity( label:String, type:String ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(type);
var entity:Entity = ObjectPool.get(Entity);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
entity.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(type, label, true, 0, 30, true);
anim.allowTransform = true;
entity.add(anim);
//assign the name
entity.id = id;
return entity;
}
```

```
public function destroyInteractEntity( entity:Entity ):Entity
{
if (entity)
{
destroyEntity(entity);
ObjectPool.give(entity.remove(Detail));
ObjectPool.give(entity.remove(Position));
ObjectPool.give(entity.remove(Animation));
if (entity.get(Grid))
ObjectPool.give(entity.remove(Grid));
}
return null;
}

[Inject]
public function set rangeBuilder( v:RangeBuilder ):void { _rangeBuilder = v; }
[Inject]
public function set routeLineBuilder( v:RouteLineBuilder ):void { _routeLineBuilder = v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }

private function get id():String
{
_id++;
return "Interact" + _id;
}
}
}

-----------------------------------
File 207: igw\com\game\entity\factory\ISectorFactory.as
package com.game.entity.factory
{
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorObjectiveData;

import org.ash.core.Entity;

public interface ISectorFactory
{
function createSectorBase( data:SectorEntityData ):Entity;

function createTransgate( data:SectorEntityData ):Entity;

function createDepot( data:SectorEntityData ):Entity;

function
```

```
createOutpost( data:SectorEntityData ):Entity;

function createDerelict( data:SectorEntityData ):Entity;

function createObjective( data:SectorObjectiveData ):Entity;

function destroySectorEntity( entity:Entity ):void;
}
}
```

File 208: igw\com\game\entity\factory\IShipFactory.as

```
package com.game.entity.factory
{
import com.service.server.incoming.data.BattleEntityData;
import com.service.server.incoming.data.SectorEntityData;

import org.ash.core.Entity;

public interface IShipFactory
{
function createShip( data:BattleEntityData ):Entity;

function createFleet( data:SectorEntityData ):Entity;

function destroyShip( ship:Entity ):void;

function destroyFleet( fleet:Entity ):void;
}
}
```

File 209: igw\com\game\entity\factory\IStarbaseFactory.as

```
package com.game.entity.factory
{
import com.game.entity.components.shared.Pylon;
import com.model.starbase.BuildingVO;
import com.service.server.incoming.data.BattleEntityData;

import org.ash.core.Entity;

public interface IStarbaseFactory
{
function createBuilding( id:String, vo:BuildingVO ):Entity;

function createBattleBuilding( data:BattleEntityData ):Entity;

function createBaseItem( id:String, vo:BuildingVO ):Entity;

function createBattleBaseItem( data:BattleEntityData ):Entity;

function
```

```
createStarbasePlatform( starbaseOwnerID:String, blend:Boolean = false ):void;

function createForcefield( key:String, pylonA:Pylon, pylonB:Pylon, color:uint ):Entity;

function updateStarbaseBuilding( entity:Entity ):void;

function createGridSquare( type:String, x:Number, y:Number ):Entity;

function createBoundingLine( startX:int, startY:int, endX:int, endY:int ):Entity;

function destroyStarbaseItem( building:Entity ):void;

function setBaseFaction(faction:String):void;
}
}

------------------------------------
File 210: igw\com\game\entity\factory\IVCFactory.as
package com.game.entity.factory
{
import com.model.starbase.BuildingVO;

import org.ash.core.Entity;

public interface IVCFactory
{
function createBuildingAnimation( buildingVO:BuildingVO, entity:Entity ):Entity;

function createBuildingConstruction( buildingVO:BuildingVO, entity:Entity ):Entity;

function createHealthBar( entity:Entity ):Entity;

function createIsoSquare( entity:Entity, type:String ):Entity;

function createName( entity:Entity, name:String ):Entity;

function createBuildingShield( buildingVO:BuildingVO, entity:Entity ):Entity;

function createShield( entity:Entity ):Entity;

function createStateBar( entity:Entity, text:String ):Entity;

function createTurret( buildingVO:BuildingVO, entity:Entity ):Entity;

function createStarbaseShield( entity:Entity ):Entity;

function createDepotCannisters( entity:Entity, index:int = 0 ):Entity;

function createDebuffTray(entity:Entity):Entity;

function
```

```
destroyComponent( component:Entity ):void;
}
}

-----------------------------------
File 211: igw\com\game\entity\factory\IVFXFactory.as
package com.game.entity.factory
{
import com.game.entity.components.battle.TrailFX;
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.DebugLineData;
import com.service.server.incoming.data.SectorBattleData;

import org.ash.core.Entity;

public interface IVFXFactory
{
function createExplosion( parent:Entity, x:int, y:int ):Entity;

function createSectorExplosion( parent:Entity, x:int, y:int ):Entity;

function createHit( hitTarget:Entity, projectile:Entity, x:int, y:int, hitShield:Boolean = false,
useProjectile:Boolean = false ):Entity;

function createAttackIcon( data:SectorBattleData ):Entity;

function createThruster( entity:Entity, attachPointProto:IPrototype, debugAttachPoints:Boolean =
false, visible:Boolean = false ):Entity;

function createMuzzle( entity:Entity, attachPointProto:IPrototype, weaponProto:IPrototype,
slotIndex:Number, visible:Boolean = false ):Entity;

function createDamageEffect( entity:Entity, attachPointProto:IPrototype ):Entity;

function createDebugLine( data:DebugLineData ):Entity;

function createTrail( trail:TrailFX, x:Number, y:Number, rotation:Number ):Entity;

function destroyAttack( attackIcon:Entity ):void;

function destroyDebugLine( debugLine:Entity ):void;

function destroyVFX( entity:Entity ):void;

function destroyTrail( trail:Entity ):void;
}
}



-----------------------------------
```

File 212: igw\com\game\entity\factory\SectorFactory.as

```
package com.game.entity.factory
{
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.sector.Mission;
import com.game.entity.components.sector.Transgate;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Cargo;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.EventComponent;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.VCList;
import com.model.asset.AssetVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.sector.SectorModel;
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorObjectiveData;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public class SectorFactory extends BaseFactory implements ISectorFactory
{
private var _playerModel:PlayerModel;
private var _sectorModel:SectorModel;

public function createSectorBase( data:SectorEntityData ):Entity
{
var type:String = _sectorModel.starbaseAsset;
var assetVO:AssetVO = _assetModel.getEntityData(type);
var sectorBase:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SECTOR, assetVO, null, data.ownerId);
detail.level = 1;
var player:PlayerVO = _playerModel.getPlayer(data.ownerId);
var level:int = (player && !player.isNPC) ? player.level : data.level;
if (level > 40)
detail.level = 5;
else
```

```
if (level > 30)
detail.level = 4;
else if (level > 20)
detail.level = 3;
else if (level > 10)
detail.level = 2;

detail.baseLevel = data.level;

detail.baseRatingTech = data.baseRatingTech;

detail.maxPlayersPerFaction = data.maxPlayersPerFaction;

sectorBase.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.STARBSE_SECTOR);
sectorBase.add(pos);
//interactive component
sectorBase.add(ObjectPool.get(Interactable));
//grid component
sectorBase.add(ObjectPool.get(Grid));
//animation component
var dmgAppend:String = data.currentHealthPct < .25 ? "DMG" : '';
var anim:Animation = ObjectPool.get(Animation);
if (data.factionPrototype != _sectorModel.sectorFaction)
{
var faction:String = (data.factionPrototype == FactionEnum.IGA) ? "IGA_" :
(data.factionPrototype == FactionEnum.SOVEREIGNTY) ? "SOV_" : "TYR_";
anim.init(assetVO.type, faction + assetVO.spriteName + detail.level, true, 0, 30, false, -6, -40);
} else
anim.init(assetVO.type, assetVO.spriteName + detail.level + dmgAppend, true, 0, 30, false, -6,
-40);
sectorBase.add(anim);
//attack component
var attack:Attack = ObjectPool.get(Attack);
attack.bubbled = data.bubbled;
sectorBase.add(attack);
//visual component list
var vcList:VCList = ObjectPool.get(VCList);
vcList.init(TypeEnum.NAME);
if (data.bubbled)
{
if (_sectorModel.sectorFaction == FactionEnum.IGA)
vcList.addComponentType(TypeEnum.STARBASE_SHIELD_IGA);
else if (_sectorModel.sectorFaction == FactionEnum.SOVEREIGNTY)
vcList.addComponentType(TypeEnum.STARBASE_SHIELD_SOVEREIGNTY);
else
vcList.addComponentType(TypeEnum.STARBASE_SHIELD_TYRANNAR);
}
sectorBase.add(vcList);
```

```
//owned or enemy
if (data.ownerId == CurrentUser.id)
sectorBase.add(ObjectPool.get(Owned));
else if (data.factionPrototype != CurrentUser.faction)
sectorBase.add(ObjectPool.get(Enemy));
//mission component
if (data.mission)
sectorBase.add(ObjectPool.get(Mission));

if (data.eventSpawn)
sectorBase.add(ObjectPool.get(EventComponent));
//assign the name
sectorBase.id = data.id;
addEntity(sectorBase);
return sectorBase;
}

public function createTransgate( data:SectorEntityData ):Entity
{
var type:String = _sectorModel.transgateAsset;
var assetVO:AssetVO = _assetModel.getEntityData(type);
var transgate:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SECTOR, assetVO);
transgate.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.STARBSE_SECTOR);
transgate.add(pos);
//interactive component
transgate.add(ObjectPool.get(Interactable));
//grid component
transgate.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, true);
anim.playing = anim.replay = false;
transgate.add(anim);
//transgagte component
var transgateC:Transgate = ObjectPool.get(Transgate);
transgateC.isPositiveWarp = data.isPositiveWarp;
transgateC.customDestinationPrototypeGroup = data.additionalInfo;
transgate.add(transgateC);
//mission component
if (data.mission)
transgate.add(ObjectPool.get(Mission));
//assign the name
transgate.id = data.id;
addEntity(transgate);
```

```
return transgate;
}

public function createDepot( data:SectorEntityData ):Entity
{
var type:String = _sectorModel.depotAsset;
var assetVO:AssetVO = _assetModel.getEntityData(type);
var depot:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SECTOR, assetVO);
depot.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.STARBSE_SECTOR);
depot.add(pos);
//interactive component
depot.add(ObjectPool.get(Interactable));
//grid component
depot.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, true);
anim.playing = anim.replay = false;
depot.add(anim);
//attack component
var attack:Attack = ObjectPool.get(Attack);
depot.add(attack);
//mission component
if (data.mission)
depot.add(ObjectPool.get(Mission));
//assign the name
depot.id = data.id;
addEntity(depot);
return depot;
}

public function createOutpost( data:SectorEntityData ):Entity
{
var type:String = _sectorModel.outpostAsset;
var assetVO:AssetVO = _assetModel.getEntityData(type);
var outpost:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SECTOR, assetVO);
outpost.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.STARBSE_SECTOR);
outpost.add(pos);
```

```
//interactive component
outpost.add(ObjectPool.get(Interactable));
//grid component
outpost.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, true);
anim.playing = anim.replay = false;
outpost.add(anim);
//attack component
var attack:Attack = ObjectPool.get(Attack);
outpost.add(attack);
//mission component
if (data.mission)
outpost.add(ObjectPool.get(Mission));
//assign the name
outpost.id = data.id;
addEntity(outpost);
return outpost;
}

public function createDerelict( data:SectorEntityData ):Entity
{
var type:String = TypeEnum.DERELICT_IGA;
if (_sectorModel.sectorFaction == FactionEnum.IMPERIUM)
type = TypeEnum.DERELICT_IMPERIUM;
else if (_sectorModel.sectorFaction == FactionEnum.SOVEREIGNTY)
type = TypeEnum.DERELICT_SOVEREIGNTY;
else if (_sectorModel.sectorFaction == FactionEnum.TYRANNAR)
type = TypeEnum.DERELICT_TYRANNAR;
var assetVO:AssetVO = _assetModel.getEntityData(type);
var derelict:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SECTOR, assetVO);
derelict.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.STARBSE_SECTOR);
derelict.add(pos);
//grid component
derelict.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, true);
anim.playing = anim.replay = false;
derelict.add(anim);
//interactive component
derelict.add(ObjectPool.get(Interactable));
//
```

```
cargo component
derelict.add(ObjectPool.get(Cargo));
//assign the name
derelict.id = data.id;
//add to the game
addEntity(derelict);
return derelict;
}

public function createObjective( data:SectorObjectiveData ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(data.asset);
var objective:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SECTOR, assetVO, null, '', 0, data.type);
objective.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.STARBSE_SECTOR);
objective.add(pos);
//grid component
objective.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, true);
anim.playing = anim.replay = false;
objective.add(anim);
//interactive component
objective.add(ObjectPool.get(Interactable));
// Mission
objective.add(ObjectPool.get(Mission));
//assign the name
objective.id = data.missionKey
//add to the game
addEntity(objective);
return objective;
}

public function destroySectorEntity( entity:Entity ):void
{
destroyEntity(entity);
ObjectPool.give(entity.remove(Detail));
ObjectPool.give(entity.remove(Position));
ObjectPool.give(entity.remove(Interactable));
ObjectPool.give(entity.remove(Grid));
ObjectPool.give(entity.remove(Animation));
if (entity.has(Attack))
ObjectPool.give(entity.remove(Attack));
if (entity.has(Cargo))
ObjectPool.give(entity.remove(Cargo));
```

```
if (entity.has(Enemy))
ObjectPool.give(entity.remove(Enemy));
if (entity.has(Mission))
ObjectPool.give(entity.remove(Mission));
if (entity.has(Owned))
ObjectPool.give(entity.remove(Owned));
if (entity.has(Transgate))
ObjectPool.give(entity.remove(Transgate));
if (entity.has(VCList))
ObjectPool.give(entity.remove(VCList));
if (entity.has(EventComponent))
ObjectPool.give(entity.remove(EventComponent));
}

[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
}
}

------------------------------------
File 213: igw\com\game\entity\factory\ShipFactory.as
package com.game.entity.factory
{
import com.enum.CategoryEnum;
import com.enum.EntityMoveEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.battle.DebuffTray;
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Modules;
import com.game.entity.components.battle.Shield;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.sector.Fleet;
import com.game.entity.components.sector.Mission;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Cargo;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.EventComponent;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.VCList;
import
```

```
com.model.asset.AssetVO;
import com.model.battle.BattleModel;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.BattleEntityData;
import com.service.server.incoming.data.ModuleData;
import com.service.server.incoming.data.SectorEntityData;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public class ShipFactory extends BaseFactory implements IShipFactory
{
private var _battleModel:BattleModel;
private var _igaLayer:int;
private var _imperiumLayer:int;
private var _malusLayer:int;
private var _playerModel:PlayerModel;
private var _tyrannarLayer:int;

[PostConstruct]
public function init():void
{
var layers:Array = [1, 2, 3];
_igaLayer = layers.splice(Math.random() * layers.length | 0, 1)[0];
_imperiumLayer = 0;
_malusLayer = layers.splice(Math.random() * layers.length | 0, 1)[0];
_tyrannarLayer = layers.splice(Math.random() * layers.length | 0, 1)[0];
}

public function createShip( data:BattleEntityData ):Entity
{
var ship:Entity = createEntity();
//assign the name
ship.id = data.id;
var prototypeVO:IPrototype = data.shipPrototype;
var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SHIP, assetVO, prototypeVO, data.ownerId);
ship.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.location.x, data.location.y, data.rotation, LayerEnum.SHIP +
getLayer(prototypeVO.getValue('faction')));
ship.add(pos);
//move component
var move:Move = ObjectPool.get(Move);
move.init(0, EntityMoveEnum.LERPING);
ship.add(move);
```

```
//health component
var health:Health = ObjectPool.get(Health);
health.init(data.currentHealth, data.maxHealth, _battleModel.getBattleEntity(data.id), .2);
ship.add(health);
//attack component
var attack:Attack = ObjectPool.get(Attack);
if (data.currentTargetId)
attack.targetID = data.currentTargetId;
ship.add(attack);
//grid component
ship.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName + '_0', true);
ship.add(anim);
//owned or enemy
if (data.ownerId == CurrentUser.id)
ship.add(ObjectPool.get(Owned));
else if (data.factionPrototype != CurrentUser.battleFaction)
ship.add(ObjectPool.get(Enemy));
//interactive component
ship.add(ObjectPool.get(Interactable));
//visual component list
var vcList:VCList = ObjectPool.get(VCList);
vcList.init(TypeEnum.HEALTH_BAR);
ship.add(vcList);
//add ship component
ship.add(ObjectPool.get(Ship));
//add a module component to keep track of the modules a ship has
var modules:Modules = ObjectPool.get(Modules);
var moduleData:ModuleData;
var proto:IPrototype;
for (var i:int = 0; i < data.modules.length; i++)
{
moduleData = data.modules[i];
if (moduleData.weaponPrototype)
{
proto = _prototypeModel.getWeaponPrototype(moduleData.weaponPrototype);
if (proto)
{
if (proto.getValue("activated") == true)
modules.addActivatedModule(moduleData.moduleIdx, proto);
else
modules.addModule(moduleData.moduleIdx, proto);
modules.addModuleByAttachPoint(moduleData.attachPointPrototype, proto);
modules.addIndexByAttachPoint(moduleData.attachPointPrototype, moduleData.moduleIdx);
}
} else if (moduleData.activeDefensePrototype)
{
proto
```

```
= _prototypeModel.getWeaponPrototype(moduleData.activeDefensePrototype);
if (proto)
{
if (proto.getValue("activated") == true)
modules.addActivatedModule(moduleData.moduleIdx, proto);
else
modules.addModule(moduleData.moduleIdx, proto);
modules.addModuleByAttachPoint(moduleData.attachPointPrototype, proto);
modules.addIndexByAttachPoint(moduleData.attachPointPrototype, moduleData.moduleIdx);
}
} else if (moduleData.modulePrototype)
{
//check to see if this ship has a shield
proto = _prototypeModel.getWeaponPrototype(moduleData.modulePrototype);
if (proto && proto.getUnsafeValue("type") == 10)
{
//add the shield component
var shield:Shield = ObjectPool.get(Shield);
shield.init(data.shieldsEnabled, data.shieldsCurrentHealth);
ship.add(shield);
if (data.shieldsEnabled)
vcList.addComponentType(TypeEnum.SHIELD);
}
if (proto)
{
modules.addModuleByAttachPoint(moduleData.attachPointPrototype, proto);
modules.addIndexByAttachPoint(moduleData.attachPointPrototype, moduleData.moduleIdx);
}
}

}
ship.add(modules);
//add to the game
addEntity(ship);
return ship;
}

public function createFleet( data:SectorEntityData ):Entity
{
var prototypeVO:IPrototype = _prototypeModel.getShipPrototype(data.shipPrototype);
if (!prototypeVO)
return null;
var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);
var fleet:Entity = createEntity();
//assign the name
fleet.id = data.id;
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.SHIP, assetVO, prototypeVO, data.ownerId);
detail.level = data.level;
detail.maxPlayersPerFaction
```

```
= data.maxPlayersPerFaction;
fleet.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.locationX, data.locationY, 1, LayerEnum.SHIP + getLayer(data.factionPrototype));
fleet.add(pos);
//move component
var move:Move = ObjectPool.get(Move);
move.init(data.travelSpeed, EntityMoveEnum.POINT_TO_POINT);
fleet.add(move);
//grid component
fleet.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName + '_0', true);
fleet.add(anim);
//attack component
var attack:Attack = ObjectPool.get(Attack);
fleet.add(attack);
//owned or enemy
if (data.ownerId == CurrentUser.id)
fleet.add(ObjectPool.get(Owned));
else if (data.factionPrototype != CurrentUser.faction || data.mission)
fleet.add(ObjectPool.get(Enemy));
//interactive component
fleet.add(ObjectPool.get(Interactable));
//visual component list
var vcList:VCList = ObjectPool.get(VCList);
vcList.init(TypeEnum.NAME);
fleet.add(vcList);
// cargo component
var cargo:Cargo = ObjectPool.get(Cargo);
cargo.cargo = data.cargo;
fleet.add(cargo);
// event component
if (data.eventSpawn)
fleet.add(ObjectPool.get(EventComponent));
//fleet component
fleet.add(ObjectPool.get(Fleet));
//mission component
if (data.mission)
fleet.add(ObjectPool.get(Mission));
//add to the game
addEntity(fleet);
return fleet;
}

public function destroyShip( ship:Entity ):void
{
destroyEntity(ship);
ObjectPool.give(ship.remove(Detail));
```

```
ObjectPool.give(ship.remove(Position));
ObjectPool.give(ship.remove(Move));
ObjectPool.give(ship.remove(Health));
if (ship.has(DebuffTray))
ObjectPool.give(ship.remove(DebuffTray));
if (ship.has(Shield))
ObjectPool.give(ship.remove(Shield));
ObjectPool.give(ship.remove(Attack));
ObjectPool.give(ship.remove(Grid));
ObjectPool.give(ship.remove(Animation));
if (ship.has(Owned))
ObjectPool.give(ship.remove(Owned));
else if (ship.has(Enemy))
ObjectPool.give(ship.remove(Enemy));
ObjectPool.give(ship.remove(Interactable));
ObjectPool.give(ship.remove(VCList));
ObjectPool.give(ship.remove(Modules));
ObjectPool.give(ship.remove(Ship));
}

public function destroyFleet( fleet:Entity ):void
{
destroyEntity(fleet);
ObjectPool.give(fleet.remove(Detail));
ObjectPool.give(fleet.remove(Position));
ObjectPool.give(fleet.remove(Move));
ObjectPool.give(fleet.remove(Grid));
ObjectPool.give(fleet.remove(Animation));
ObjectPool.give(fleet.remove(Interactable));
ObjectPool.give(fleet.remove(Fleet));
ObjectPool.give(fleet.remove(VCList));
if (fleet.has(Owned))
ObjectPool.give(fleet.remove(Owned));
if (fleet.has(Enemy))
ObjectPool.give(fleet.remove(Enemy));
if (fleet.has(Mission))
ObjectPool.give(fleet.remove(Mission));
if (fleet.has(EventComponent))
ObjectPool.give(fleet.remove(EventComponent));
}

private function getLayer( faction:String ):int
{
switch (faction)
{
case FactionEnum.IGA:
return _igaLayer;
case FactionEnum.SOVEREIGNTY:
return _malusLayer;
case
```

```
FactionEnum.IMPERIUM:
return _imperiumLayer;
}
return _tyrannarLayer;
}

[Inject]
public function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
}
}



---
File 214: igw\com\game\entity\factory\StarbaseFactory.as
package com.game.entity.factory
{
import com.Application;
import com.controller.transaction.TransactionController;
import com.enum.AudioEnum;
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.event.StateEvent;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Shield;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Pylon;
import com.game.entity.components.shared.VCList;
import com.game.entity.components.shared.fsm.FSM;
import com.game.entity.components.shared.fsm.Forcefield;
import com.game.entity.components.shared.fsm.TurretFSM;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.Platform;
import com.game.entity.components.starbase.State;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.asset.AssetVO;
import com.model.battle.BattleModel;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import
```

```
com.model.prototype.IPrototype;
import com.model.prototype.PrototypeVO;
import com.model.sector.SectorModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.service.server.incoming.data.BattleEntityData;
import com.util.AllegianceUtil;
import com.util.CommonFunctionUtil;
import com.util.InteractEntityUtil;

import flash.geom.Point;

import org.ash.core.Entity;
import org.greensock.TweenManual;
import org.shared.ObjectPool;

public class StarbaseFactory extends BaseFactory implements IStarbaseFactory
{
private var _battleModel:BattleModel;
private var _cornerVO:BuildingVO;
private var _i:int;
private var _point:Point = new Point();
private var _playerModel:PlayerModel;
private var _sectorModel:SectorModel;
private var _starbaseModel:StarbaseModel;
private var _starbaseVO:BuildingVO;
private var _tempBuildingVO:BuildingVO = new BuildingVO();
private var _transactionController:TransactionController;
private var _baseFaction:String = FactionEnum.IGA;

public function createBuilding( id:String, vo:BuildingVO ):Entity
{
var faction:String = CurrentUser.id == CurrentUser.id ? CurrentUser.faction :
_playerModel.getPlayer(CurrentUser.id).faction;

var building:Entity = createEntity();
var prototypeVO:IPrototype = vo.prototype;

/*
var assetName:String = prototypeVO.asset + '_' + FactionEnum.getFactionShort(faction);
var assetVO:AssetVO = _assetModel.getEntityData(assetName);

if( assetVO == null)
assetVO = _assetModel.getEntityData(prototypeVO.asset);
*/

var assetVO:AssetVO = _assetModel.getEntityData(prototypeVO.asset);

/**/
if(assetVO.spriteSheetsString == "Buildings")
{
```

```
if(assetVO.spriteXML.length > 0)
{
//assetVO.spriteSheetsString = 'Buildings_TYR';
assetVO.sprites[0] = 'sprite/Buildings_' + FactionEnum.getFactionShort(faction) + '.png';
assetVO.spriteXML[0] = 'sprite/Buildings_' + FactionEnum.getFactionShort(faction) + '.xml';
//assetVO.spriteSheetsString += '_' + FactionEnum.getFactionShort(faction);
//assetVO.spriteXML[0] += '_' + FactionEnum.getFactionShort(faction);
//assetVO.spriteSheetsString = 'TyrannarBuildings';
//assetVO.setOneSpriteXML('sprite/TyrannarBuildings.xml');

//assetVO.spriteSheetsString = faction + "Buildings";
//assetVO.spriteSheetsString = 'TyrannarBuildings';
//assetVO.setSpriteXML(0, 'sprite/TyrannarBuildings.xml');
//assetVO.setSpriteXML(0, "sprite/" + faction + "Buildings.xml");
//assetVO.spriteXML[0] = "sprite/" + faction + "Buildings.xml";
}
}

if(assetVO.spriteSheetsString == "BaseWeaponsA")
{
if(assetVO.spriteXML.length > 0)
{
assetVO.sprites[0] = 'sprite/BaseWeaponsA_' + FactionEnum.getFactionShort(faction) + '.png';
assetVO.spriteXML[0] = 'sprite/BaseWeaponsA_' + FactionEnum.getFactionShort(faction) +
'.xml';
}
}
//*/

//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.BUILDING, assetVO, prototypeVO, CurrentUser.id);
building.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(0, 0, 0, LayerEnum.BUILDING);
_starbaseModel.grid.convertBuildingGridToIso(pos.position, vo);
building.add(pos);
//grid component
building.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, '', true);
building.add(anim);
//interactive component
building.add(ObjectPool.get(Interactable));
//owned
building.add(ObjectPool.get(Owned));
//building component
var
```

```
buildingC:Building = ObjectPool.get(Building);
buildingC.init(vo);
buildingC.faction = faction;
building.add(buildingC);
//vclist component
var vcList:VCList = ObjectPool.get(VCList);
vcList.init();
building.add(vcList);

//additional components based on itemClass
if (vo.itemClass == TypeEnum.PYLON)
{
var pylon:Pylon = ObjectPool.get(Pylon);
pylon.baseX = vo.baseX;
pylon.baseY = vo.baseY;
pylon.color = AllegianceUtil.instance.getFactionColor(CurrentUser.faction);
building.add(pylon);
pylon.bottom = createPylonPlatform(vo, pos);
} else if (vo.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM)
{
var turretFSM:TurretFSM = ObjectPool.get(TurretFSM);
var fsm:FSM = ObjectPool.get(FSM);
fsm.init(turretFSM);
building.add(fsm);
}
//assign the name
building.id = id;
building.add(faction);
//set the building animation
updateStarbaseBuilding(building);
//add to the game
addEntity(building);
return building;
}

public function createBattleBuilding( data:BattleEntityData ):Entity
{
var faction:String = data.ownerId == CurrentUser.id ? CurrentUser.faction :
_playerModel.getPlayer(data.ownerId).faction;

var building:Entity = createEntity();
_tempBuildingVO.prototype = data.buildingPrototype;
_tempBuildingVO.baseX = data.gridLocationX;
_tempBuildingVO.baseY = data.gridLocationY;
_tempBuildingVO.currentHealth = data.currentHealth;
for (_i = 0; _i < data.modules.length; _i++)
{
if (data.modules[_i].modulePrototype)

_tempBuildingVO.equipModule(_prototypeModel.getWeaponPrototype(data.modules[_i].modulePrototype)
data.modules[_i].slotPrototype);
```

```
else if (data.modules[_i].weaponPrototype)

_tempBuildingVO.equipModule(_prototypeModel.getWeaponPrototype(data.modules[_i].weaponPrototype
data.modules[_i].slotPrototype);
}

/*var assetName:String = _tempBuildingVO.asset + '_' + FactionEnum.getFactionShort(faction);
var assetVO:AssetVO = _assetModel.getEntityData(assetName);

if( assetVO == null)
assetVO = _assetModel.getEntityData(_tempBuildingVO.asset);*/
var assetVO:AssetVO = _assetModel.getEntityData(_tempBuildingVO.asset);
if(assetVO.spriteSheetsString == "Buildings")
{
if(assetVO.spriteXML.length > 0)
{
assetVO.sprites[0] = 'sprite/Buildings_' + FactionEnum.getFactionShort(faction) + '.png';
assetVO.spriteXML[0] = 'sprite/Buildings_' + FactionEnum.getFactionShort(faction) + '.xml';
}
}
if(assetVO.spriteSheetsString == "BaseWeaponsA")
{
if(assetVO.spriteXML.length > 0)
{
assetVO.sprites[0] = 'sprite/BaseWeaponsA_' + FactionEnum.getFactionShort(faction) + '.png';
assetVO.spriteXML[0] = 'sprite/BaseWeaponsA_' + FactionEnum.getFactionShort(faction) +
'.xml';
}
}

//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.BUILDING, assetVO, _tempBuildingVO.prototype, data.ownerId);
building.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(data.gridLocationX, data.gridLocationY, 0, LayerEnum.BUILDING);
_starbaseModel.grid.convertBuildingGridToIso(pos.position, _tempBuildingVO);
building.add(pos);
//grid component
building.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, '', true);
building.add(anim);

var oid:String = CurrentUser.id;
//owned or enemy
if (data.ownerId == CurrentUser.id)
{
```

```
building.add(ObjectPool.get(Owned));
if (_tempBuildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM &&
data.currentHealth > 0)
building.add(ObjectPool.get(Interactable));
} else if (data.factionPrototype != CurrentUser.battleFaction)
{
if (data.currentHealth > 0)
building.add(ObjectPool.get(Interactable));
building.add(ObjectPool.get(Enemy));
}
//health component
var health:Health = ObjectPool.get(Health);
health.init(data.currentHealth, data.maxHealth, _battleModel.getBattleEntity(data.id), 1);
building.add(health);
//building component
var buildingC:Building = ObjectPool.get(Building);
buildingC.init(_tempBuildingVO);
buildingC.faction = faction;
building.add(buildingC);
//shield component
var shield:Shield = ObjectPool.get(Shield);
shield.init(data.shieldsEnabled, data.shieldsCurrentHealth, true);
building.add(shield);
//vclist component
var vcList:VCList = ObjectPool.get(VCList);
if (data.currentHealth > 0)
vcList.init(TypeEnum.HEALTH_BAR);
else
vcList.init();
if (data.shieldsEnabled && data.currentHealth > 0)
{
if (_tempBuildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM)
vcList.addComponentType(TypeEnum.TURRET_SHIELD);
else
vcList.addComponentType(TypeEnum.BUILDING_SHIELD);
}
building.add(vcList);
//additional components based on itemClass
if (_tempBuildingVO.itemClass == TypeEnum.PYLON)
{
var pylon:Pylon = ObjectPool.get(Pylon);
pylon.baseX = _tempBuildingVO.baseX;
pylon.baseY = _tempBuildingVO.baseY;
pylon.color =
AllegianceUtil.instance.getFactionColor(_playerModel.getPlayer(data.ownerId).faction);
building.add(pylon);
pylon.bottom = createPylonPlatform(_tempBuildingVO, pos);
} else if (_tempBuildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM)
{
//attack
```

```
component
var attack:Attack = ObjectPool.get(Attack);
if (data.currentTargetId)
attack.targetID = data.currentTargetId;
building.add(attack);
}
//assign the name
building.id = data.id;
building.add(faction);
//set the building animation
updateStarbaseBuilding(building);
//add to the game
if (buildingC.buildingVO)
addEntity(building);
_tempBuildingVO = new BuildingVO();
return building;
}

public function createBaseItem( id:String, vo:BuildingVO ):Entity
{
if (vo.constructionCategory == "Wall")
return null;

var baseItem:Entity = createEntity();
var prototypeVO:IPrototype = vo.prototype;
var assetVO:AssetVO = getBaseItemAssetVO(prototypeVO);

//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, _assetModel.getEntityData(vo.asset), prototypeVO);
baseItem.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(0, 0, 1, LayerEnum.STARBSE_SECTOR);
_starbaseModel.grid.convertBuildingGridToIso(pos.position, vo);
baseItem.add(pos);
//grid component
baseItem.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, '');
baseItem.add(anim);
//interactive component
baseItem.add(ObjectPool.get(Interactable));
//platform component
var platform:Platform = ObjectPool.get(Platform);
platform.buildingVO = vo;
baseItem.add(platform);
//assign the name
baseItem.id = id;
//set
```

```
the building animation
updateStarbaseBuilding(baseItem);
//add to the game
addEntity(baseItem);
return baseItem;
}

public function createBattleBaseItem( data:BattleEntityData):Entity
{
_tempBuildingVO.prototype = data.buildingPrototype;
_tempBuildingVO.baseX = data.gridLocationX;
_tempBuildingVO.baseY = data.gridLocationY;
var assetVO:AssetVO = getBaseItemAssetVO(_tempBuildingVO.prototype);
var baseItem:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, _assetModel.getEntityData(_tempBuildingVO.asset),
_tempBuildingVO.prototype, data.ownerId);
baseItem.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(0, 0, 1, LayerEnum.STARBSE_SECTOR);
_starbaseModel.grid.convertBuildingGridToIso(pos.position, _tempBuildingVO);
baseItem.add(pos);
//grid component
baseItem.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, '');
baseItem.add(anim);
//platform component
var platform:Platform = ObjectPool.get(Platform);
platform.buildingVO = _tempBuildingVO;
_tempBuildingVO = new BuildingVO();
baseItem.add(platform);
//assign the name
baseItem.id = data.id;
//set the building animation
updateStarbaseBuilding(baseItem);
//add to the game
addEntity(baseItem);
return baseItem;
}

public function createStarbasePlatform( starbaseOwnerID:String, blend:Boolean = false ):void
{
var level:int = starbaseOwnerID == CurrentUser.id ? CurrentUser.level :
_playerModel.getPlayer(starbaseOwnerID).level;
var faction:String = starbaseOwnerID == CurrentUser.id ? CurrentUser.faction :
_playerModel.getPlayer(starbaseOwnerID).faction;
if
```

```
(level >= 10 && !_game.getEntity("StarbaseBaseB"))
createStarbaseExtension("StarbaseBaseB", "StarbaseBaseB", blend, faction);
if (level >= 20 && !_game.getEntity("StarbaseBaseC"))
createStarbaseExtension("StarbaseBaseC", "StarbaseBaseC", blend, faction);
if (level >= 30 && !_game.getEntity("StarbaseBaseD"))
createStarbaseExtension("StarbaseBaseD", "StarbaseBaseD", blend, faction);
if (level >= 40 && !_game.getEntity("StarbaseBaseE"))
createStarbaseExtension("StarbaseBaseE", "StarbaseBaseE", blend, faction);
//create the corner
if (!_cornerVO)
{
_cornerVO = new BuildingVO();
_cornerVO.baseX = 268;
_cornerVO.baseY = 274;
_cornerVO.prototype = new PrototypeVO({itemClass:'', sizeX:5, sizeY:5});
}
if (!_game.getEntity("StarbaseCorner"))
createStarbasePart('StarbaseCorner', 'Corner', _cornerVO, faction);
//create the base
if (!_starbaseVO)
{
_starbaseVO = new BuildingVO();
_starbaseVO.baseX = 275;
_starbaseVO.baseY = 275;
_starbaseVO.prototype = new PrototypeVO({itemClass:'', sizeX:45, sizeY:45});
}
if (!_game.getEntity("StarbasePlatform"))
createStarbasePart('StarbasePlatform', 'StarbaseBaseA', _starbaseVO, faction);
}

public function createForcefield( key:String, pylonA:Pylon, pylonB:Pylon, color:uint ):Entity
{
var baseX:int = pylonA.baseX < pylonB.baseX ? pylonA.baseX : pylonB.baseX;
if (pylonA.baseX != pylonB.baseX)
baseX += 5;
var baseY:int = pylonA.baseY < pylonB.baseY ? pylonA.baseY : pylonB.baseY;
if (pylonA.baseY != pylonB.baseY)
baseY += 5;
var dir:String = (pylonA.baseX == pylonB.baseX) ? "Left" : "Right";
var sizeX:int = Math.abs(pylonA.baseX - pylonB.baseX) - 5;
if (sizeX <= 0)
sizeX = 5;
var sizeY:int = Math.abs(pylonA.baseY - pylonB.baseY) - 5;
if (sizeY <= 0)
sizeY = 5;

var asset:AssetVO = _assetModel.getEntityData(TypeEnum.FORCEFIELD);
var building:Building;
var detail:Detail;
var field:Forcefield;
var
```

```
forcefield:Entity = _game.getEntity(key);
var pos:Position;
var prototype:IPrototype = createforceFieldPrototype(sizeX, sizeY);
if (forcefield)
{
building = forcefield.get(Building);
building.buildingVO.baseX = baseX;
building.buildingVO.baseY = baseY;
building.buildingVO.prototype = prototype;
detail = forcefield.get(Detail);
detail.prototypeVO = prototype;
pos = forcefield.get(Position);
_starbaseModel.grid.convertBuildingGridToIso(pos.position, building.buildingVO);
pos.dirty = true;
field = Forcefield(FSM(forcefield.get(FSM)).component);
field.adjustFieldLengths();
} else
{
_tempBuildingVO.prototype = prototype;
_tempBuildingVO.baseX = baseX;
_tempBuildingVO.baseY = baseY;
_tempBuildingVO.currentHealth = 100;
forcefield = createEntity();
// Add Detail component
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.BUILDING, asset, prototype);
forcefield.add(detail);
// Add Animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(asset.type, asset.spriteName + dir, true, 0, 30, true, 0, 51);
anim.alpha = 0;
forcefield.add(anim);
// Add Position component
pos = ObjectPool.get(Position);
pos.init(0, 0, 0, LayerEnum.BUILDING);
_starbaseModel.grid.convertBuildingGridToIso(pos.position, _tempBuildingVO);
forcefield.add(pos);
// Building component
building = ObjectPool.get(Building);
building.init(_tempBuildingVO);
forcefield.add(building);
// Add Grid component
//forcefield.add(ObjectPool.get(Grid));
// Add Forcefield component
field = ObjectPool.get(Forcefield);
field.animation = anim;
field.color = color;
field.building = _tempBuildingVO;
// Add FSM component
var fsm:FSM = ObjectPool.get(FSM);
fsm.init(field);
```

```
forcefield.add(fsm);
forcefield.id = key;
// Add the thruster to the game
_game.addEntity(forcefield);
_tempBuildingVO = new BuildingVO();
_soundController.playSound(AudioEnum.AFX_BARRIER_UP, 0.5);
}
return forcefield;
}

public function updateStarbaseBuilding( entity:Entity ):void
{
if (entity)
{
setBuildingAnimation(entity);
}
}

public function createGridSquare( type:String, x:Number, y:Number ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(type);
var isoSquare:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, assetVO);
isoSquare.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(type, assetVO.spriteName, false);
anim.color = 0xffff00;
isoSquare.add(anim);
//position component
var newPos:Position = ObjectPool.get(Position);
newPos.init(x, y, 0, 6);
isoSquare.add(newPos);
//grid component
isoSquare.add(ObjectPool.get(Grid));
//assign the name
isoSquare.id = x + type + y;
//add to game
addEntity(isoSquare);
return isoSquare;
}

public function createBoundingLine( startX:int, startY:int, endX:int, endY:int ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.DEBUG_LINE);
if (!assetVO)
{
_assetModel.addGameAssetData(InteractEntityUtil.createPrototype(TypeEnum.DEBUG_LINE,
```

```
4, "DebugLine"));
assetVO = _assetModel.getEntityData(TypeEnum.DEBUG_LINE);
}

var boundingLine:Entity = createEntity();
// Add Detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(assetVO.type, assetVO);
boundingLine.add(detail);

// Add Animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, false, 0, 30, true);
var xDiff:Number = (startX - endX) * (startX - endX);
var yDiff:Number = (startY - endY) * (startY - endY);
anim.scaleX = Math.sqrt(xDiff + yDiff) / 64;
anim.scaleY = 1;
anim.color = AllegianceUtil.instance.getFactionColor(CurrentUser.faction);
anim.allowTransform = true;
anim.visible = true;
boundingLine.add(anim);

// Add Position component
var position:Position = ObjectPool.get(Position);
position.init(startX, startY, 0, LayerEnum.RANGE);
position.rotation = Math.atan2(endY - startY, endX - startX);
boundingLine.add(position);

boundingLine.id = startX + "." + startY + "." + endX + "." + endY;
// Add the debug line to the game
addEntity(boundingLine);
return boundingLine;
}

public function destroyStarbaseItem( building:Entity ):void
{
if (!building)
return;
destroyEntity(building);
ObjectPool.give(building.remove(Detail));
ObjectPool.give(building.remove(Position));
if (building.has(Grid))
ObjectPool.give(building.remove(Grid));
ObjectPool.give(building.remove(Animation));
if (building.has(Interactable))
ObjectPool.give(building.remove(Interactable));
if (building.has(Building))
{
var b:Building = building.remove(Building);
if (b.buildingVO.itemClass == TypeEnum.FORCEFIELD)
_soundController.playSound(AudioEnum.AFX_BARRIER_DOWN,
```

```
0.5);
ObjectPool.give(b);
}
if (building.has(Platform))
ObjectPool.give(building.remove(Platform));
if (building.has(VCList))
ObjectPool.give(building.remove(VCList));
if (building.has(State))
ObjectPool.give(building.remove(State));
if (building.has(Owned))
ObjectPool.give(building.remove(Owned));
if (building.has(Enemy))
ObjectPool.give(building.remove(Enemy));
if (building.has(Health))
ObjectPool.give(building.remove(Health));
if (building.has(Attack))
ObjectPool.give(building.remove(Attack));
if (building.has(Shield))
ObjectPool.give(building.remove(Shield));
if (building.has(Pylon))
{
var pylon:Pylon = building.remove(Pylon);
destroyStarbaseItem(pylon.bottom);
ObjectPool.give(pylon);
}
if (building.has(FSM))
ObjectPool.give(building.remove(FSM));
}
public function setBaseFaction(faction:String):void
{
_baseFaction = faction;
}

private function createStarbasePart( id:String, label:String, vo:BuildingVO, faction:String =
FactionEnum.IGA ):Entity
{
var type:String = TypeEnum.STARBASE_IGA;
if (faction != FactionEnum.IGA)
type = (faction == FactionEnum.SOVEREIGNTY) ? TypeEnum.STARBASE_SOVEREIGNTY :
TypeEnum.STARBASE_TYRANNAR;
var baseItem:Entity = createEntity();
var assetVO:AssetVO = _assetModel.getEntityData(type);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, assetVO);
baseItem.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(2500, 2500, 1, LayerEnum.STARBSE_SECTOR);
baseItem.add(pos);
//grid
```

```
component
baseItem.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(type, label, false, 0, 30, true, 712, 303);
baseItem.add(anim);
//platform component
var platform:Platform = ObjectPool.get(Platform);
platform.buildingVO = vo;
baseItem.add(platform);
//assign the name
baseItem.id = id;
addEntity(baseItem);
return baseItem;
}

private function createStarbaseExtension( id:String, label:String, blend:Boolean = false,
faction:String = FactionEnum.IGA ):Entity
{
var type:String = TypeEnum.STARBASE_IGA;
if (faction != FactionEnum.IGA)
type = (faction == FactionEnum.SOVEREIGNTY) ? TypeEnum.STARBASE_SOVEREIGNTY :
TypeEnum.STARBASE_TYRANNAR;
var baseItem:Entity = createEntity();
var assetVO:AssetVO = _assetModel.getEntityData(type);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, assetVO);
baseItem.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(2500, 2500, 1, LayerEnum.BACKGROUND);
baseItem.add(pos);
//grid component
baseItem.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(type, label, false, 0, 30, true, 712, 303);
baseItem.add(anim);
if (blend)
{
anim.alpha = 0;
TweenManual.to(anim, 3, {alpha:1});
}
//assign the name
baseItem.id = id;
addEntity(baseItem);
return baseItem;
}

private
```

```
function setBuildingAnimation( entity:Entity ):void
{
var animation:Animation = entity.get(Animation);
var vo:BuildingVO = (entity.has(Building)) ? Building(entity.get(Building)).buildingVO :
Platform(entity.get(Platform)).buildingVO;

/*var assetVO:AssetVO = null;

if(entity.has(Building) && Building(entity.get(Building)).faction.length > 0)
{
var assetName:String = vo.asset + '_' +
FactionEnum.getFactionShort(Building(entity.get(Building)).faction);

assetVO = _assetModel.getEntityData(assetName);

if( assetVO == null)
assetVO = _assetModel.getEntityData(vo.asset);
else
assetVO.type = vo.asset;
}
else
assetVO = _assetModel.getEntityData(vo.asset);
*/
/**/

var faction:String = CurrentUser.id == CurrentUser.id ? CurrentUser.faction :
_playerModel.getPlayer(CurrentUser.id).faction;

var assetVO:AssetVO = _assetModel.getEntityData(vo.asset);
/*if(assetVO.spriteSheetsString == "Buildings")
{
if(assetVO.spriteXML.length > 0)
{
//entity.get(Building)).faction
assetVO.sprites[0] = 'sprite/Buildings_' + + FactionEnum.getFactionShort(faction) + '.png';
assetVO.spriteXML[0] = 'sprite/Buildings_' + + FactionEnum.getFactionShort(faction) + '.xml';
}
}
if(assetVO.spriteSheetsString == "BaseWeaponsA")
{
if(assetVO.spriteXML.length > 0)
{
assetVO.sprites[0] = 'sprite/BaseWeaponsA_' + FactionEnum.getFactionShort(faction) + '.png';
assetVO.spriteXML[0] = 'sprite/BaseWeaponsA_' + FactionEnum.getFactionShort(faction) +
'.xml';
}
}
//*/

var vcList:VCList = entity.get(VCList);
var
```

```
level:int = CommonFunctionUtil.getBuildingVisualLevel(vo.level);

//set the correct animation ie. normal, damaged, constructing etc.
switch (vo.itemClass)
{
case TypeEnum.COMMAND_CENTER:
case TypeEnum.CONSTRUCTION_BAY:
case TypeEnum.DOCK:
case TypeEnum.DEFENSE_DESIGN:
case TypeEnum.ADVANCED_TECH:
case TypeEnum.SHIPYARD:
case TypeEnum.REACTOR_STATION:
case TypeEnum.ACADEMY:
case TypeEnum.OFFICERS_LOUNGE:
case TypeEnum.WEAPONS_FACILITY:
case TypeEnum.SURVEILLANCE:
case TypeEnum.SHIELD_GENERATOR:
if (vo.currentHealth == 0)
{
if (animation.type != TypeEnum.BUILDINGS_DAMAGED)
animation.type = TypeEnum.BUILDINGS_DAMAGED;
animation.label = assetVO.spriteName + "Destroyed" + level;
vo.damaged = vo.destroyed = true;
} else if (vo.currentHealth < StarbaseSystem.BUILDING_DAMAGED_HEALTH)
{
if (animation.type != TypeEnum.BUILDINGS_DAMAGED)
animation.type = TypeEnum.BUILDINGS_DAMAGED;
animation.label = assetVO.spriteName + "Damaged" + level;
vo.destroyed = false;
vo.damaged = true;
} else
{
if (animation.type != vo.itemClass)
animation.type = vo.itemClass;
animation.label = assetVO.spriteName + level;
vo.damaged = vo.destroyed = false;
}

//add the building animation
if (assetVO.type == TypeEnum.REACTOR_STATION || assetVO.type ==
TypeEnum.SHIELD_GENERATOR || assetVO.type == TypeEnum.SURVEILLANCE)
{
(vo.currentHealth > 0) ? vcList.addComponentType(TypeEnum.BUILDING_ANIMATION) :
vcList.removeComponentType(TypeEnum.BUILDING_ANIMATION);
}
break;
case TypeEnum.PYLON:
var pylon:Pylon = entity.get(Pylon);
var bottom:Entity = pylon.bottom;
if (vo.currentHealth == 0)
{
```

```
//pylon
animation.alpha = 0;
if (animation.type != TypeEnum.BUILDINGS_DAMAGED)
animation.type = TypeEnum.BUILDINGS_DAMAGED;
animation.label = assetVO.spriteName + "Damaged" + level;
//pylon base
animation = bottom.get(Animation);
animation.label = "BarrierDestroyed" + level;
vo.damaged = vo.destroyed = true;
} else if (vo.currentHealth < StarbaseSystem.BUILDING_DAMAGED_HEALTH)
{
if (animation.type != TypeEnum.BUILDINGS_DAMAGED)
animation.type = TypeEnum.BUILDINGS_DAMAGED;
animation.label = assetVO.spriteName + "Damaged" + level;
animation.alpha = 1;
vo.destroyed = false;
vo.damaged = true;
//pylon base
assetVO = Detail(bottom.get(Detail)).assetVO;
animation = bottom.get(Animation);
animation.label = assetVO.spriteName + "Damaged" + level;
vcList.addComponentType(TypeEnum.BUILDING_ANIMATION);
} else
{
if (animation.type != vo.itemClass)
animation.type = vo.itemClass;
animation.alpha = 1;
animation.label = assetVO.spriteName + level;
vo.damaged = vo.destroyed = false;
//pylon base
assetVO = Detail(bottom.get(Detail)).assetVO;
animation = bottom.get(Animation);
animation.label = assetVO.spriteName + level;
vcList.addComponentType(TypeEnum.BUILDING_ANIMATION);
}
break;
case TypeEnum.RESOURCE_DEPOT:
case TypeEnum.POINT_DEFENSE_PLATFORM:
if (vo.currentHealth == 0)
{
if (animation.type != TypeEnum.BUILDINGS_DAMAGED)
animation.type = TypeEnum.BUILDINGS_DAMAGED;
animation.label = assetVO.spriteName + "Destroyed";
vo.damaged = vo.destroyed = true;
} else if (vo.currentHealth < StarbaseSystem.BUILDING_DAMAGED_HEALTH)
{
if (animation.type != TypeEnum.BUILDINGS_DAMAGED)
animation.type = TypeEnum.BUILDINGS_DAMAGED;
animation.label = assetVO.spriteName + "Damaged";
vo.destroyed
```

```
= false;
vo.damaged = true;
} else
{
if (animation.type != vo.itemClass)
animation.type = vo.itemClass;
animation.label = assetVO.spriteName;
vo.damaged = vo.destroyed = false;
}
//add the turret animation
if (assetVO.type == TypeEnum.POINT_DEFENSE_PLATFORM)
{
(vo.currentHealth > 0) ? vcList.addComponentType(TypeEnum.STARBASE_TURRET) :
vcList.removeComponentType(TypeEnum.STARBASE_TURRET);
}
//add/remove canisters from resource depots
if (assetVO.type == TypeEnum.RESOURCE_DEPOT)
(vo.currentHealth > 0) ?
vcList.addComponentType(TypeEnum.RESOURCE_DEPOT_CANISTER) :
vcList.removeComponentType(TypeEnum.RESOURCE_DEPOT_CANISTER);
break;
case TypeEnum.STARBASE_WALL:
animation.label = assetVO.type;
animation.offsetX = 66.625;
animation.offsetY = 32.825;
break;
case TypeEnum.STARBASE_ARM:
animation.label = assetVO.type;
animation.offsetX = 66.625;
animation.offsetY = 32.825;
break;
case TypeEnum.STARBASE_PLATFORMA:
animation.label = assetVO.type;
animation.offsetX = 194;
animation.offsetY = 97.5;
break;
case TypeEnum.STARBASE_PLATFORMB:
animation.label = assetVO.type;
animation.offsetX = 194;
animation.offsetY = 97.5;
break;
default:
_point.setTo(0, 0);
break;
}
//remove the shield if the building is destroyed
if (vcList && vo.currentHealth == 0)
{
vcList.removeComponentType(TypeEnum.BUILDING_SHIELD);
vcList.removeComponentType(TypeEnum.TURRET_SHIELD);
if
```

```
(vcList.hasComponentType(TypeEnum.HEALTH_BAR))
vcList.removeComponentType(TypeEnum.HEALTH_BAR);
}
}

private function createPylonPlatform( vo:BuildingVO, position:Position):Entity
{
var assetVO:AssetVO = getBaseItemAssetVO(vo.prototype);
var baseItem:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, assetVO);
baseItem.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(position.x, position.y, 1, LayerEnum.STARBSE_SECTOR);
baseItem.add(pos);
//grid component
baseItem.add(ObjectPool.get(Grid));
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, '', true);
baseItem.add(anim);
//platform component
var buildingVO:BuildingVO = ObjectPool.get(BuildingVO);
buildingVO.prototype = vo.prototype;
buildingVO.baseX = vo.baseX;
buildingVO.baseY = vo.baseY;
var platform:Platform = ObjectPool.get(Platform);
platform.buildingVO = buildingVO;
baseItem.add(platform);
//assign the name
baseItem.id = vo.id + "Base";
//add to the game
addEntity(baseItem);
return baseItem;
}

private function getBaseItemAssetVO( prototype:IPrototype ):AssetVO
{
var type:String = '';
switch (prototype.itemClass)
{
case TypeEnum.PYLON:
type = TypeEnum.PYLON_BASE_IGA;
if (baseFaction != FactionEnum.IGA)
type = (baseFaction == FactionEnum.SOVEREIGNTY) ?
TypeEnum.PYLON_BASE_SOVEREIGNTY : TypeEnum.PYLON_BASE_TYRANNAR;
break;
case TypeEnum.STARBASE_ARM:
type
```

```
= TypeEnum.STARBASE_ARM_IGA;
if (baseFaction != FactionEnum.IGA)
type = (baseFaction == FactionEnum.SOVEREIGNTY) ?
TypeEnum.STARBASE_ARM_SOVEREIGNTY : TypeEnum.STARBASE_ARM_TYRANNAR;
break;
case TypeEnum.STARBASE_PLATFORMA:
type = TypeEnum.STARBASE_PLATFORMA_IGA;
if (baseFaction != FactionEnum.IGA)
type = (baseFaction == FactionEnum.SOVEREIGNTY) ?
TypeEnum.STARBASE_PLATFORMA_SOVEREIGNTY :
TypeEnum.STARBASE_PLATFORMA_TYRANNAR;
break;
case TypeEnum.STARBASE_PLATFORMB:
type = TypeEnum.STARBASE_PLATFORMB_IGA;
if (baseFaction != FactionEnum.IGA)
type = (baseFaction == FactionEnum.SOVEREIGNTY) ?
TypeEnum.STARBASE_PLATFORMB_SOVEREIGNTY :
TypeEnum.STARBASE_PLATFORMB_TYRANNAR;
break;
}
return _assetModel.getEntityData(type);
}

private function createforceFieldPrototype( sizeX:int, sizeY:int ):IPrototype
{
return new PrototypeVO({
"key":"Forcefield",
"asset":"Forcefield",
"uiAsset":"Forcefield",
"sizeX":sizeX,
"sizeY":sizeY,
"health":50000,
"itemClass":"Forcefield"});
}

private function get baseFaction():String
{
switch (Application.STATE)
{
case StateEvent.GAME_BATTLE_INIT:
case StateEvent.GAME_BATTLE:
case StateEvent.GAME_SECTOR_INIT:
case StateEvent.GAME_SECTOR:
return _baseFaction;
//return _sectorModel.sectorFaction;
}
return CurrentUser.faction;
}

[Inject]
public
```

```
function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
}
}

------------------------------------
File 215: igw\com\game\entity\factory\VCFactory.as
package com.game.entity.factory
{
import com.Application;
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.event.StateEvent;
import com.game.entity.components.battle.DebuffTray;
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Shield;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Pylon;
import com.game.entity.components.shared.fsm.FSM;
import com.game.entity.components.shared.fsm.TurretFSM;
import com.game.entity.components.shared.render.Render;
import com.game.entity.components.shared.render.RenderSprite;
import com.game.entity.components.shared.render.RenderSpriteStarling;
import com.game.entity.components.shared.render.RenderStarling;
import com.game.entity.components.shared.render.VisualComponent;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.Construction;
import com.game.entity.components.starbase.State;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.asset.AssetVO;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.model.sector.SectorModel;
import com.model.starbase.BuildingVO;
import com.util.AllegianceUtil;
import com.util.CommonFunctionUtil;

import
```

```
flash.display.BlendMode;
import flash.geom.Point;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public class VCFactory extends BaseFactory implements IVCFactory
{
//values for sector name tags (TODO: better spot for this stuff?)
public static const FONT_COLOR:uint = 0xffffff;
public static const FONT_SIZE:int = 14;
public static const TEXT_WIDTH:int = 462;
public static const TEXT_HEIGHT:int = 60;
public static const TEXT_WIDTH_HALF:int = int(TEXT_WIDTH / 2);
public static const TEXT_HEIGHT_HALF:int = int(TEXT_HEIGHT / 2);

private static const DEPOTX:Array = [175, 155, 124, 93, 74, 74, 93, 124, 155, 175];
private static const DEPOTY:Array = [127, 141, 145, 141, 127, 111, 97, 92, 97, 111];

private var _id:int = 0;
private var _point:Point = new Point();
private var _sectorModel:SectorModel;
private var _playerModel:PlayerModel;

public function createBuildingAnimation( buildingVO:BuildingVO, entity:Entity ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.BUILDING_ANIMATION);
var buildingAnimation:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
buildingAnimation.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
buildingAnimation.add(anim);
var render:IRender = createAndAddRender(buildingAnimation, entity, 0, 0, 1, true);
var level:int = CommonFunctionUtil.getBuildingVisualLevel(buildingVO.level);
//if constructing is true then we want to show the construction animation
switch (buildingVO.asset)
{
case TypeEnum.PYLON:
anim.init(assetVO.type, 'BarrierGlow' + level, true, 0, 25, true);
render.x = render.y = 152;
anim.color = render.color = Pylon(entity.get(Pylon)).color;
break;
case TypeEnum.REACTOR_STATION:
anim.init(assetVO.type, 'Turbine' + level, true, 0, 25, true);
render.x = render.y = 125;
break;
case TypeEnum.SHIELD_GENERATOR:
anim.init(assetVO.type,
```

```
'Electric' + level, true, 0, 10, true);
render.x = render.y = 100;
break;
case TypeEnum.SURVEILLANCE:
anim.init(assetVO.type, 'Dish5', false, 0, 12, true);
anim.scaleX = anim.scaleY = .5 + level * .1;
if (level < 5)
{
anim.scaleX -= .1;
anim.scaleY -= .1;
}
render.scaleX = anim.scaleX;
render.scaleY = anim.scaleX;
IRender(render).x = IRender(render).y = 125 - (125 * anim.scaleY);
anim.allowTransform = true;
break;
}
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
buildingAnimation.add(vc);
//assign the name
buildingAnimation.id = id;
addEntity(buildingAnimation);
return buildingAnimation;
}

/**
* Creates a construction animation that appears above a building
* To create this animation, we need to actually make three entities.
* 1 is the beam of light, 2 is the "mothership", 3 is the glow on the mothership
* @param buildingVO The buildingVO that we are constructing
* @param entity The entity that we are placing this animation on
* @return Our newly created construction entity
*/
public function createBuildingConstruction( buildingVO:BuildingVO, entity:Entity ):Entity
{
var parentBuilding:Building = entity.get(Building);
//components
var animation:Animation;
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.BUILDING_CONSTRUCTION);
var detail:Detail;
var position:Position;
var render:IRender;
var visualComponent:VisualComponent;

//create the beam of light
var beam:Entity = createEntity();
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
beam.add(detail);
```

```
animation = ObjectPool.get(Animation);
animation.init(assetVO.type, 'ConstructBeam0', false, 0, 30, true);
animation.alpha = 0;
beam.add(animation);
beam.id = id;

//create the second beam of light
var beam2:Entity = createEntity();
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
beam2.add(detail);
animation = ObjectPool.get(Animation);
animation.init(assetVO.type, 'ConstructBeam1', false, 0, 30, true);
animation.alpha = 0;
beam2.add(animation);
beam2.id = id;

//create the glow
var glow:Entity = createEntity();
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
glow.add(detail);
animation = ObjectPool.get(Animation);
animation.init(assetVO.type, 'ConstructGlow', false, 0, 30, true);
animation.alpha = 0;
glow.add(animation);
glow.id = id;

//create the "mothership"
var mothership:Entity = createEntity();
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
mothership.add(detail);
animation = ObjectPool.get(Animation);
animation.init(assetVO.type, assetVO.spriteName, false, 0, 30, true);
animation.alpha = 0;
mothership.add(animation);
var construction:Construction = ObjectPool.get(Construction);
construction.beam = beam;
construction.beam2 = beam2;
construction.glow = glow;
construction.mothership = mothership;
construction.owner = entity;

visualComponent = ObjectPool.get(VisualComponent);
visualComponent.init(entity);
mothership.add(visualComponent);
mothership.id = id;

//fsm
```

```
var fsm:FSM = ObjectPool.get(FSM);
fsm.init(construction);
mothership.add(fsm);

//lots of hardcoded numbers. YAYAYAYAYAY!!
var scale:Number = (buildingVO.sizeX == 15) ? 1 : (buildingVO.sizeY == 10) ? .73 :
(parentBuilding.buildingVO.itemClass == TypeEnum.PYLON) ? .62 : .45;
var size:Number = (buildingVO.sizeX == 15) ? 400 : (buildingVO.sizeY == 10) ? 250 :
(parentBuilding.buildingVO.itemClass == TypeEnum.PYLON) ? 300 : 200;
var x:Number = (size - (258 * scale)) / 2;
construction.ypos = (buildingVO.sizeX == 15) ? 0 : (buildingVO.sizeY == 10) ? -25 :
(parentBuilding.buildingVO.itemClass == TypeEnum.PYLON) ? 15 : 0;

var state:State = entity.get(State);
if (state && state.transaction)
{
if (state.transaction.timeMS - state.transaction.timeRemainingMS >= 500)
construction.state = Construction.RESTABILIZE;
}

render = createAndAddRender(beam, entity, x, construction.ypos);
render.scaleX = scale;
render.scaleY = .5;
render.alpha = 0;
render = createAndAddRender(beam2, entity, x, construction.ypos);
render.scaleX = scale;
render.scaleY = .5;
render.alpha = 0;
construction.beamScaleY = 1 * scale;
render = createAndAddRender(mothership, entity, x, construction.ypos);
render.scaleX = render.scaleY = scale;
render.alpha = 0;
render = createAndAddRender(glow, entity, x, construction.ypos);
render.scaleX = render.scaleY = scale;
render.alpha = 0;
render.alpha = 0;

addEntity(beam);
addEntity(beam2);
addEntity(glow);
addEntity(mothership);
return mothership;
}

public function createHealthBar( entity:Entity ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.HEALTH_BAR);
var currentHealth:Health = entity.get(Health);
var health:Entity = createEntity();
//detail
```

```
component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
health.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(TypeEnum.HEALTH_BAR, assetVO.spriteName, true, 0, 30, true, 0, 40);
anim.allowTransform = true;
health.add(anim);
//health component
var healthComponent:Health = entity.get(Health);
healthComponent.animation = anim;
health.add(healthComponent);
anim.scaleX = healthComponent.percent;
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.VFX;
newPos.ignoreRotation = true;
oldPos.addLink(newPos);
health.add(newPos);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
health.add(vc);
//assign the name
health.id = id;
//add to game
addEntity(health);
return health;
}

public function createIsoSquare( entity:Entity, type:String ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(type);
var isoSquare:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.STARBASE, assetVO);
isoSquare.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(type, assetVO.spriteName, true, 0, 30, true);
anim.color = 0xffff00;
isoSquare.add(anim);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.MISC;
oldPos.addLink(newPos);
isoSquare.add(newPos);
```

```
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
isoSquare.add(vc);
//assign the name
isoSquare.id = id;
//add to game
addEntity(isoSquare);
return isoSquare;
}

public function createStateBar( entity:Entity, text:String ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.STATE_BAR);
var state:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
state.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(TypeEnum.STATE_BAR, assetVO.spriteName, true, 0, 30, true, 0, 40);
anim.allowTransform = true;
anim.text = text;
state.add(anim);
//add to the entities state component
var stateC:State = entity.get(State);
stateC.component = state;
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.VFX;
oldPos.addLink(newPos);
state.add(newPos);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
state.add(vc);
//assign the name
state.id = id;
//add to game
addEntity(state);
return state;
}

public function createBuildingShield( buildingVO:BuildingVO, entity:Entity ):Entity
{
var shield:Entity = createEntity();
var faction:String = _sectorModel.sectorFaction;
var
```

```
buildingDetail:Detail = entity.get(Detail);
if (buildingDetail)
var buildingOwner:PlayerVO = _playerModel.getPlayer(buildingDetail.ownerID);

if (buildingOwner)
faction = buildingOwner.faction;

//detail component
var assetVO:AssetVO = (buildingVO.prototype.itemClass !=
TypeEnum.POINT_DEFENSE_PLATFORM) ?
_assetModel.getEntityData(TypeEnum.BUILDING_SHIELD) :
_assetModel.getEntityData(TypeEnum.TURRET_SHIELD);
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
shield.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
var shieldName:String = (buildingVO.sizeX == 5) ? "BuildingShieldSmall" : ((buildingVO.sizeX
== 10) ? "BuildingShieldMedium" : "BuildingShieldLarge");
anim.init(assetVO.type, shieldName, true, 0, 30, true);
anim.playing = anim.replay = false;
anim.alpha = .5;

if (faction == FactionEnum.IMPERIUM)
anim.color = 0x00ff00;
if (faction == FactionEnum.IGA)
anim.color = 0x6bd7ff;
else if (faction == FactionEnum.SOVEREIGNTY)
anim.color = 0xc96bff;
else
anim.color = 0xff7d4f;
shield.add(anim);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.BUILDING;
newPos.depth = oldPos.depth + 1;
oldPos.addLink(newPos);
shield.add(newPos);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
shield.add(vc);
//assign the name
shield.id = id;
addEntity(shield);
return shield;
}

public function createShield( entity:Entity ):Entity
{
```

```
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.SHIELD);
var shield:Entity = createEntity();

//detail component
var shipDetail:Detail = entity.get(Detail);
var faction:String = shipDetail.prototypeVO.getValue('faction');
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
shield.add(detail);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.SHIELD;
oldPos.addLink(newPos);
shield.add(newPos);
//Scaling
var shipAssetVO:AssetVO = _assetModel.getEntityData(shipDetail.prototypeVO.asset);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(TypeEnum.SHIELD, assetVO.spriteName, true, 0, 30, true);
anim.playing = anim.replay = false;
if (faction == FactionEnum.IGA)
anim.color = 0x5263F6;
else if (faction == FactionEnum.SOVEREIGNTY)
anim.color = 0x9652F6;
else if (faction == FactionEnum.TYRANNAR)
anim.color = 0xFF4500;
else
anim.color = 0x00ff00;
anim.blendMode = BlendMode.ADD;
anim.scaleX = anim.scaleY = shipAssetVO.shieldScale;
anim.allowTransform = true;
shield.add(anim);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
shield.add(vc);
//add the shield
var shieldC:Shield = entity.get(Shield);
shieldC.animation = anim;
shield.add(shieldC);
//assign the name
shield.id = id;
//add to game
addEntity(shield);
return shield;
}

public function createStarbaseShield( entity:Entity ):Entity
{
```

```
var entityDetail:Detail = entity.get(Detail);
var type:String = _sectorModel.starbaseShieldAsset;
var assetVO:AssetVO = _assetModel.getEntityData(type);
var starbaseShield:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
starbaseShield.add(detail);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.RANGE;
newPos.rotation = 0;
newPos.dirty = false;
oldPos.addLink(newPos);
starbaseShield.add(newPos);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(type, assetVO.spriteName, true, 0, 30, true);
//anim.color = AllegianceUtil.instance.getFactionColor(_sectorModel.sectorFaction);
anim.blendMode = BlendMode.ADD;
switch (entityDetail.level)
{
case 1:
case 2:
anim.offsetX = 3;
anim.offsetY = 4;
anim.scaleX = .8;
anim.scaleY = .6;
break;
case 3:
anim.offsetX = 3;
anim.offsetY = -4;
anim.scaleX = .8;
anim.scaleY = .6;
break;
case 4:
anim.offsetX = 3;
anim.offsetY = -23;
anim.scaleX = .8;
anim.scaleY = .7;
break;
case 5:
anim.offsetX = 2;
anim.offsetY = -32;
anim.scaleX = .9;
anim.scaleY = .9;
break;
}
anim.allowTransform
```

```
= true;
starbaseShield.add(anim);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
starbaseShield.add(vc);
//assign the name
starbaseShield.id = id;
//add to game
addEntity(starbaseShield);
return starbaseShield;
}

public function createName( entity:Entity, name:String ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.NAME);
var text:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO, null, Detail(entity.get(Detail)).ownerID);
text.add(detail);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.TEXT;
oldPos.addLink(newPos);
text.add(newPos);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(TypeEnum.NAME, '', true, 0, 30, true, TEXT_WIDTH_HALF, TEXT_HEIGHT_HALF);
anim.playing = anim.replay = false;
anim.text = name;
anim.color = AllegianceUtil.instance.getEntityColor(entity);
anim.forceReady();
text.add(anim);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
text.add(vc);
//assign the name
text.id = id;
//add to game
addEntity(text);
return text;
}

public function createTurret( buildingVO:BuildingVO, entity:Entity ):Entity
{
//TODO: Not a problem at the moment since all turrets only have one slot but may need to
extend this in the future to handle multiple slots
var
```

```
slot:String = buildingVO.getValue("slots")[0];
var module:IPrototype = (buildingVO.modules.hasOwnProperty(slot)) ? buildingVO.modules[slot]
: null;
if (module == null)
return null;
var turret:Entity = createEntity();
//detail component
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.STARBASE_TURRET);
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
turret.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
var level:int = module.getValue('level');
if (level > 5)
level = 5;

switch (module.asset)
{
case TypeEnum.TURRET_BEAM_POINT_DEFENSE_CLUSTER:
anim.init(TypeEnum.STARBASE_TURRET, 'CC' + level, true, 0, 30, true);
_point.setTo(104, 29);
break;
case TypeEnum.TURRET_DRONE:
anim.init(TypeEnum.STARBASE_TURRET, 'DB' + level, true, 0, 30, true);
_point.setTo(106, 32);
break;
case TypeEnum.TURRET_BOMBARDMENT_CANNON:
anim.init(TypeEnum.STARBASE_TURRET, 'BC' + level, true, 0, 30, true);
_point.setTo(108, 34);
break;
case TypeEnum.TURRET_SENTINEL_MOUNT:
anim.init(TypeEnum.STARBASE_TURRET, 'SC' + level, true, 0, 30, true);
_point.setTo(104, 31);
break;
case TypeEnum.TURRET_MISSILE_POD:
anim.init(TypeEnum.STARBASE_TURRET, 'MP' + level, true, 0, 30, true);
_point.setTo(108, 29);
break;
}
anim.playing = anim.replay = false;
turret.add(anim);
//add the turret animation into the turret finite state machine
if (Application.STATE == StateEvent.GAME_STARBASE)
TurretFSM(FSM(entity.get(FSM)).component).animation = anim;
//add the render
createAndAddRender(turret, entity, _point.x, _point.y, 1, true);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
turret.add(vc);
```

```
//assign the name
turret.id = id;
addEntity(turret);
return turret;
}

public function createDepotCannisters( entity:Entity, index:int = 0 ):Entity
{
var assetVO:AssetVO =
_assetModel.getEntityData(TypeEnum.RESOURCE_DEPOT_CANISTER);
var building:Building = entity.get(Building);
var system:StarbaseSystem = StarbaseSystem(_game.getSystem(StarbaseSystem));
var percent:Number = (building.buildingVO.percentFilled > 0) ?
building.buildingVO.percentFilled * 100 : 0;

var cannisters:Entity = createEntity();
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
cannisters.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName + "_" + percent, false, 0, 30, true);
cannisters.add(anim);
createAndAddRender(cannisters, entity, DEPOTX[index] - 12, DEPOTY[index] - 42, 1, true);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
cannisters.add(vc);
if (index <= 9)
vc.addChild(createDepotCannisters(entity, index + 1));
//assign the name
cannisters.id = id;
addEntity(cannisters);
return cannisters;
}

private function createAndAddRender( entity:Entity, target:Entity, x:Number = 0, y:Number = 0,
depth:int = -1, takeParentsColor:Boolean = false ):IRender
{
//get the animation component of the building
var eAnimation:Animation = entity.get(Animation);
var tAnimation:Animation = target.get(Animation);
//add the render
var render:IRender;
if (Application.STARLING_ENABLED)
{
render = IRender(ObjectPool.get(RenderStarling));
if (depth == -1 || depth >= RenderSpriteStarling(tAnimation.render).numChildren)
RenderSpriteStarling(tAnimation.render).addChild(RenderStarling(render));
```

```
else
RenderSpriteStarling(tAnimation.render).addChildAt(RenderStarling(render), depth);
} else
{
render = IRender(ObjectPool.get(Render));
if (depth == -1 || depth >= RenderSprite(tAnimation.render).numChildren)
RenderSprite(tAnimation.render).addChild(Render(render));
else
RenderSprite(tAnimation.render).addChildAt(Render(render), depth);
}
if (takeParentsColor && tAnimation.color != 0 && tAnimation.color != 0xffffff)
render.color = tAnimation.color;
render.x = x;
render.y = y;
eAnimation.render = render;
return render;
}

public function createDebuffTray( entity:Entity ):Entity
{
var debuffTrayComponent:DebuffTray = entity.get(DebuffTray);
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.DEBUFF_TRAY);
var tray:Entity = createEntity();
tray.add(debuffTrayComponent);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.DEBUFF, assetVO);
tray.add(detail);
//animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(TypeEnum.HEALTH_BAR, "", true, 0, 30, true, 0, 32);
tray.add(anim);
debuffTrayComponent.Draw(anim, assetVO);
//position component
var oldPos:Position = Position(entity.get(Position));
var newPos:Position = oldPos.clone();
newPos.layer = LayerEnum.VFX;
newPos.ignoreRotation = true;
oldPos.addLink(newPos);
tray.add(newPos);
//visual component
var vc:VisualComponent = ObjectPool.get(VisualComponent);
vc.init(entity);
tray.add(vc);
//assign the name
tray.id = id;
//add to game
addEntity(tray);
return tray;
}
```

```
private function removeRender( entity:Entity, render:IRender = null ):void
{
if (entity == null && render == null)
return;
var animation:Animation = entity.get(Animation);
if (entity && render == null)
render = animation.render;
//remove the render from the building
if (Application.STARLING_ENABLED)
{
var rs:RenderStarling = RenderStarling(render);
rs.removeFromParent();
ObjectPool.give(rs);
} else
{
var r:Render = Render(render);
r.parent.removeChild(r);
ObjectPool.give(r);
}
}

public function destroyComponent( component:Entity ):void
{
if (!component)
return;
var detail:Detail = Detail(component.get(Detail));
if (!detail)
return;
destroyEntity(component);
switch (detail.type)
{
case TypeEnum.HEALTH_BAR:
var health:Health = component.remove(Health);
health.animation = null;
break;
case TypeEnum.DEBUFF_TRAY:
case TypeEnum.ATTACK_DEBUFF:
case TypeEnum.DEFENSE_DEBUFF:
case TypeEnum.SPEED_DEBUFF:
var tray:DebuffTray = component.remove(DebuffTray);
tray.Draw(null);
break;
case TypeEnum.SHIELD:
var shield:Shield = component.remove(Shield);
shield.animation = null;
break;
case TypeEnum.STATE_BAR:
var state:State = VisualComponent(component.get(VisualComponent)).parent.get(State);
if
```

```
(state)
state.component = null;
break;
case TypeEnum.BUILDING_ANIMATION:
//remove the render from the building
removeRender(component);
break;
case TypeEnum.STARBASE_TURRET:
var vc:VisualComponent = component.get(VisualComponent);
if (vc)
{
//remove the animation component from the turret finite state machine
if (Application.STATE == StateEvent.GAME_STARBASE)
TurretFSM(FSM(vc.parent.get(FSM)).component).animation = null;
}
//remove the render from the building
removeRender(component);
break;
case TypeEnum.BUILDING_CONSTRUCTION:
if (component.has(VisualComponent))
{
removeRender(component);
var construction:Construction = Construction(FSM(component.get(FSM)).component);
removeRender(construction.beam);
removeRender(construction.beam2);
removeRender(construction.glow);
destroyComponent(construction.beam);
destroyComponent(construction.beam2);
destroyComponent(construction.glow);
}
break;
case TypeEnum.RESOURCE_DEPOT_CANISTER:
var visualComponent:VisualComponent =
VisualComponent(component.get(VisualComponent));
while (visualComponent.numChildren > 0)
{
removeRender(component);
destroyComponent(visualComponent.removeChildAt(0));
}
break;
}
ObjectPool.give(component.remove(Detail));
ObjectPool.give(component.remove(Animation));
if (component.has(VisualComponent))
ObjectPool.give(component.remove(VisualComponent));
if (component.has(Position))
ObjectPool.give(component.remove(Position));
if (component.has(FSM))
ObjectPool.give(component.remove(FSM));
}

private
```

```
function get id():String
{
_id++;
return "vc" + _id;
}

[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }

[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
}
}

-----------------------------------
File 216: igw\com\game\entity\factory\VFXFactory.as
package com.game.entity.factory
{
import com.Application;
import com.enum.AudioEnum;
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.LayerEnum;
import com.enum.TypeEnum;
import com.event.StateEvent;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.battle.Damage;
import com.game.entity.components.battle.TrailFX;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.DebugLine;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Muzzle;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Thruster;
import com.model.asset.AssetVO;
import com.model.battle.BattleModel;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.DebugLineData;
import com.service.server.incoming.data.SectorBattleData;
import com.util.BattleUtils;

import flash.display.BlendMode;

import org.adobe.utils.StringUtil;
import org.ash.core.Entity;
import org.parade.core.IViewStack;
import org.parade.enum.PlatformEnum;
import
```

```
org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class VFXFactory extends BaseFactory implements IVFXFactory
{
private var _attackFactory:IAttackFactory;
private var _battleModel:BattleModel;
private var _explosionScale:Array = [0.25, 0.5, 1];
private var _id:int = 1;
private var _hitCounter:int = 0;
private var _viewStack:IViewStack;
private var _playerModel:PlayerModel;

public function createExplosion( parent:Entity, x:int, y:int ):Entity
{
var animation:Animation = Animation(parent.get(Animation));
var detail:Detail = Detail(parent.get(Detail));
var itemClass:String = detail.prototypeVO ? detail.prototypeVO.itemClass : '';
if (itemClass == TypeEnum.FORCEFIELD)
return null;
if (itemClass == '' && (detail.type == TypeEnum.STARBASE_SECTOR_IGA ||
detail.type == TypeEnum.STARBASE_SECTOR_SOVEREIGNTY ||
detail.type == TypeEnum.STARBASE_SECTOR_TYRANNAR))
itemClass = TypeEnum.STARBASE_SECTOR_TYRANNAR;
var expScale:Number = 1.0;
var audioFile:String;
if (animation.visible)
{
var assetVO:AssetVO;
var showShockwave:int;
var showFlare:int;
switch (itemClass)
{
//small explosion
case TypeEnum.FIGHTER:
case TypeEnum.HEAVY_FIGHTER:
case TypeEnum.SHIELD_GENERATOR:
case TypeEnum.POINT_DEFENSE_PLATFORM:
case TypeEnum.PYLON:
assetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_SMALL);
expScale = 0.75;
audioFile = (detail.category == CategoryEnum.BUILDING) ?
AudioEnum.AFX_BLD_EXPLOSION_SMALL : AudioEnum.AFX_SHIP_EXPLOSION_SMALL;
if (detail.category)
{
expScale = 1;
_viewStack.shake(1, 1, 6, 3);
}
break;
//large explosion
case
```

```
TypeEnum.BATTLESHIP:
case TypeEnum.DREADNOUGHT:
case TypeEnum.TRANSPORT:
showShockwave = Math.floor((Math.random() * 2)) + 1;
showFlare = Math.floor((Math.random() * 2)) + 1;
if (showShockwave == 1)
createShockwave(detail, animation, 5, x, y);
if (showFlare == 1)
createFlare(detail, animation, 1, x, y);
if (Application.STATE == StateEvent.GAME_BATTLE)
_viewStack.shake(5, 2, 10, 3);
case TypeEnum.COMMAND_CENTER:
case TypeEnum.CONSTRUCTION_BAY:
case TypeEnum.DOCK:
assetVO = DeviceMetrics.PLATFORM == PlatformEnum.MOBILE ?
_assetModel.getEntityData(TypeEnum.EXPLOSION_SMALL) :
_assetModel.getEntityData(TypeEnum.EXPLOSION_LARGE);
audioFile = (detail.category == CategoryEnum.BUILDING) ?
AudioEnum.AFX_BLD_EXPLOSION_BIG : AudioEnum.AFX_SHIP_EXPLOSION_BIG;
if (detail.category == CategoryEnum.BUILDING)
{
_viewStack.shake(5, 2, 10, 3);
}
break;
case "DroneBay":
case "AssaultSquadron":
case "BombardierWing":
case "DroneSquadron":
assetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_SMALL);
expScale = 0.20;
audioFile = AudioEnum.AFX_SHIP_EXPLOSION_SMALL;
break;
//medium explosion
default:
showShockwave = Math.floor((Math.random() * 2)) + 1;
showFlare = Math.floor((Math.random() * 2)) + 1;
if (showShockwave == 1)
createShockwave(detail, animation, 1, x, y);
if (showFlare == 1)
createFlare(detail, animation, 1, x, y);

assetVO = DeviceMetrics.PLATFORM == PlatformEnum.MOBILE ?
_assetModel.getEntityData(TypeEnum.EXPLOSION_SMALL) :
_assetModel.getEntityData(TypeEnum.EXPLOSION_LARGE);
expScale = 0.8;
audioFile = (detail.category == CategoryEnum.BUILDING) ?
AudioEnum.AFX_BLD_EXPLOSION_MEDIUM :
AudioEnum.AFX_SHIP_EXPLOSION_MEDIUM;
if (detail.category == CategoryEnum.BUILDING)
{
expScale
```

```
= 1;
_viewStack.shake(3, 1.3, 8, 3);
}
break;
}
var explosion:Entity = createEntity();
//detail component
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.EXPLOSION, assetVO);
explosion.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(x, y, 1, LayerEnum.EXPLOSION);
pos.rotation = Math.random() * 360;
pos.depth = 0;
explosion.add(pos);
//animation component
animation = ObjectPool.get(Animation);
animation.init(assetVO.type, assetVO.spriteName, true, 0, 30, false);
animation.scaleX = animation.scaleY = expScale;
animation.allowTransform = true;
animation.destroyOnComplete = true;
explosion.add(animation);
//grid component
explosion.add(ObjectPool.get(Grid));
//assign the name
explosion.id = name;
//add to game
addEntity(explosion);
//Play Explosion
_soundController.playSound(audioFile, 0.5);
return explosion;
}
return null;
}

public function createSectorExplosion( parent:Entity, x:int, y:int ):Entity
{
var animation:Animation = Animation(parent.get(Animation));
if (!animation.visible)
return null;
var detail:Detail = Detail(parent.get(Detail));
var itemClass:String = detail.prototypeVO ? detail.prototypeVO.itemClass : '';
if (itemClass == '')
itemClass = detail.type;
var expScale:Number = 1.0;
var audioFile:String;
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_SMALL);
var showShockwave:int;
var showFlare:int;
switch
```

```
(itemClass)
{
case TypeEnum.FIGHTER:
case TypeEnum.HEAVY_FIGHTER:
expScale = 0.75;
audioFile = AudioEnum.AFX_SHIP_EXPLOSION_SMALL;
break;

case TypeEnum.STARBASE_SECTOR_IGA:
case TypeEnum.STARBASE_SECTOR_SOVEREIGNTY:
case TypeEnum.STARBASE_SECTOR_TYRANNAR:
audioFile = AudioEnum.AFX_BLD_EXPLOSION_BIG;
_viewStack.shake(5, 2, 10, 3);
createShockwave(detail, animation, 5, x, y);
break;

case TypeEnum.BATTLESHIP:
case TypeEnum.DREADNOUGHT:
case TypeEnum.TRANSPORT:
showShockwave = Math.floor((Math.random() * 2)) + 1;
showFlare = Math.floor((Math.random() * 2)) + 1;
if (showShockwave == 1)
createShockwave(detail, animation, 3, x, y);
if (showFlare == 1)
createFlare(detail, animation, 1, x, y);
if (Application.STATE == StateEvent.GAME_BATTLE)
_viewStack.shake(5, 2, 10, 3);
audioFile = AudioEnum.AFX_SHIP_EXPLOSION_BIG;
break;

case TypeEnum.DERELICT_IGA:
case TypeEnum.DERELICT_SOVEREIGNTY:
case TypeEnum.DERELICT_TYRANNAR:
createFlare(detail, animation, .75, x, y);
return null;

default:
showFlare = Math.floor((Math.random() * 2)) + 1;
if (showFlare == 1)
createFlare(detail, animation, 1, x, y);
expScale = 0.8;
audioFile = AudioEnum.AFX_SHIP_EXPLOSION_MEDIUM;
break;
}
var explosion:Entity = createEntity();
//detail component
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.EXPLOSION, assetVO);
explosion.add(detail);
//position component
var
```

```
pos:Position = ObjectPool.get(Position);
pos.init(x, y, 1, LayerEnum.EXPLOSION);
pos.rotation = Math.random() * 360;
pos.depth = 0;
explosion.add(pos);
//animation component
animation = ObjectPool.get(Animation);
animation.init(assetVO.type, assetVO.spriteName, true, 0, 30, false);
animation.scaleX = animation.scaleY = expScale;
animation.allowTransform = true;
animation.destroyOnComplete = true;
explosion.add(animation);
//grid component
explosion.add(ObjectPool.get(Grid));
//assign the name
explosion.id = name;
//add to game
addEntity(explosion);
//Play Explosion
_soundController.playSound(audioFile, 0.5);
return explosion;
return null;
}

private function createShockwave( detail:Detail, animation:Animation, expScale:Number, x:int,
y:int ):void
{
var shockVO:AssetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_SHOCKWAVE);
var explosion:Entity = createEntity();
//detail component
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.EXPLOSION, shockVO);
explosion.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(x, y, 1, LayerEnum.EXPLOSION);
pos.depth = 2;
explosion.add(pos);
//animation component
animation = ObjectPool.get(Animation);
animation.init(TypeEnum.EXPLOSION_SHOCKWAVE, shockVO.spriteName, true, 0, 30, false);
animation.scaleX = animation.scaleY = expScale;
animation.allowTransform = true;
animation.destroyOnComplete = true;
explosion.add(animation);
//grid component
explosion.add(ObjectPool.get(Grid));
//assign the name
explosion.id = name;
//add to game
addEntity(explosion);
```

```
}

private function createFlare( detail:Detail, animation:Animation, expScale:Number, x:int, y:int
):void
{
var shockVO:AssetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_FLARE);
var explosion:Entity = createEntity();
//detail component
detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.EXPLOSION, shockVO);
explosion.add(detail);
//position component
var pos:Position = ObjectPool.get(Position);
pos.init(x, y, 0, LayerEnum.EXPLOSION);
pos.depth = 1;
explosion.add(pos);
//animation component
animation = ObjectPool.get(Animation);
animation.init(TypeEnum.EXPLOSION_FLARE, shockVO.spriteName, true, 0, 30, false);
animation.scaleX = animation.scaleY = expScale;
animation.blendMode = BlendMode.ADD;
animation.allowTransform = true;
animation.destroyOnComplete = true;
explosion.add(animation);
//grid component
explosion.add(ObjectPool.get(Grid));
//assign the name
explosion.id = name;
//add to game
addEntity(explosion);
}

public function createHit( hitTarget:Entity, projectile:Entity, x:int, y:int, hitShield:Boolean = false,
useProjectile:Boolean = false ):Entity
{
if (hitTarget == null || projectile == null)
return null;

var detail:Detail = Detail(projectile.get(Detail));
var shakeIt:Boolean = detail.prototypeVO.getValue('shakeOnHit');

_hitCounter++;
if (_hitCounter < 3 && !shakeIt)
return null;
_hitCounter = 0;

var animation:Animation = Animation(projectile.get(Animation));

var targetDetail:Detail = Detail(hitTarget.get(Detail));
var
```

```
hitItemClass:String = targetDetail.prototypeVO ? targetDetail.prototypeVO.itemClass : '';

if (!animation.visible || detail.prototypeVO == null)
return null;

if (useProjectile)
_attackFactory.cleanAttack(projectile);

var itemClass:String = detail.prototypeVO.getValue('itemClass');
var assetVO:AssetVO;


//detail component
detail = (useProjectile) ? projectile.get(Detail) : ObjectPool.get(Detail);
if (hitShield)
assetVO = _assetModel.getEntityData(TypeEnum.SHIELD_HIT);
else if (itemClass == 'PlasmaMissile' || itemClass == 'MissileBattery' || itemClass == 'MissilePod'
|| itemClass == 'AntimatterTorpedo' || itemClass == 'GravitonBomb')
assetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_MISSILEHIT);
else if (itemClass == 'PulseLaser' || itemClass == 'DisintegrationRay' || itemClass ==
'GravitonBeam')
assetVO = _assetModel.getEntityData(TypeEnum.EXPLOSION_LASERHIT);
else
assetVO = _assetModel.getEntityData(TypeEnum.HIT);
detail.init(CategoryEnum.EXPLOSION, assetVO);
//position component
var pos:Position = useProjectile ? projectile.get(Position) : ObjectPool.get(Position);
pos.init(x, y, 1, LayerEnum.HIT);
//animation component
animation = useProjectile ? projectile.get(Animation) : ObjectPool.get(Animation);
animation.init(assetVO.type, assetVO.spriteName, true, 0, 30, true);
if (hitShield)
animation.color = _battleModel.baseFactionColor;
//Scale the hit based on ship type
var hitScale:Number = 0;
//This only applicable to lasers
switch (hitItemClass)
{
case TypeEnum.FIGHTER:
case TypeEnum.HEAVY_FIGHTER:
hitScale = 0.15;
break;
case TypeEnum.CORVETTE:
case TypeEnum.DESTROYER:
hitScale = 0.18;
break;
case TypeEnum.BATTLESHIP:
case TypeEnum.DREADNOUGHT:
case TypeEnum.TRANSPORT:
hitScale = 0.2;
break;
```

```
default:
hitScale = 0;
}

if (hitScale == 0)
{
switch (itemClass)
{
case 'ParticleBlaster':
case 'StrikeCannon':
hitScale = _explosionScale[Math.floor((Math.random() * 2))];
break;
case 'Railgun':
hitScale = _explosionScale[Math.floor((Math.random() * 2))];
break;
case 'PlasmaMissile':
case 'MissileBattery':
case 'MissilePod':
hitScale = _explosionScale[Math.floor((Math.random() * 2))];
break;
default:
hitScale = 0.5;
}

}
animation.scaleX = animation.scaleY = hitScale;
animation.allowTransform = true;
animation.destroyOnComplete = true;

// Do screen shake if appropriate
if (shakeIt)
_viewStack.shake(5, 2, 17, 3);

if (!useProjectile)
{
var hit:Entity = createEntity();
hit.add(detail);
hit.add(pos);
hit.add(animation);
hit.id = name;
addEntity(hit);
} else
{
animation.spritePack = null;
animation.forceSprite = null;
_game.updateEntityID(projectile, name);
pos.layerSwap(LayerEnum.ATTACK, LayerEnum.HIT);
}
return (useProjectile) ? projectile : hit;
}
```

```
public function createAttackIcon( data:SectorBattleData ):Entity
{
var attackIcon:Entity = createEntity();
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.ATTACK_ICON);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
attackIcon.add(detail);
//animation component
var animation:Animation = ObjectPool.get(Animation);
animation.init(assetVO.type, assetVO.spriteName, true, 0, 30, false, 0, -30);
animation.allowTransform = true;
animation.scaleX = animation.scaleY = 0.9;
attackIcon.add(animation);
//grid component
attackIcon.add(ObjectPool.get(Grid));
//position component
var position:Position = ObjectPool.get(Position);
position.init(data.locationX, data.locationY - 32, 0, LayerEnum.VFX);
attackIcon.add(position);
//attack component
var attack:Attack = ObjectPool.get(Attack);
attack.attackData = data;
attackIcon.add(attack);
//assign the name
attackIcon.id = data.id;
//add to game
addEntity(attackIcon);
return attackIcon;
}

public function createTrail( trail:TrailFX, x:Number, y:Number, rotation:Number ):Entity
{
var missileTrail:Entity = createEntity();
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.MISSILE_TRAIL);
//detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, assetVO);
missileTrail.add(detail);
//animation component
var animation:Animation = ObjectPool.get(Animation);
animation.init(assetVO.type, assetVO.spriteName, false, 0, 30, true);
animation.color = trail.color;
animation.allowTransform = true;
missileTrail.add(animation);
//position component
var position:Position = ObjectPool.get(Position);
position.init(x, y, rotation, LayerEnum.VFX);
missileTrail.add(position);
```

```
//assign the name
missileTrail.id = name;
//add to game
addEntity(missileTrail);
return missileTrail;
}

public function createThruster( entity:Entity, attachPointProto:IPrototype,
debugAttachPoints:Boolean = false, visible:Boolean = false ):Entity
{
var thrusterAsset:AssetVO = _assetModel.getEntityData(TypeEnum.THRUSTER);

if(thrusterAsset == null)
return null;

if(attachPointProto == null)
return null;

// Add Detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, thrusterAsset);
detail.prototypeVO = attachPointProto;

// Add Animation component
var ownerId:String = Detail(entity.get(Detail)).ownerID;
var player:PlayerVO = _playerModel.getPlayer(ownerId);
if(player == null)
return null;

var ownerFaction:String = player.faction;
var factionPrepend:String = "SOV/";
if (ownerFaction == FactionEnum.IGA)
factionPrepend = "IGA/";
else if (ownerFaction == FactionEnum.TYRANNAR)
factionPrepend = "TYR/";
else if (ownerFaction == FactionEnum.IMPERIUM)
factionPrepend = "IMP/";
var anim:Animation = ObjectPool.get(Animation);
anim.init(thrusterAsset.type, factionPrepend + thrusterAsset.spriteName, false, 0, 30, visible);
//PR: Turning the blendmode off on these for now as it causes more of a performance hit and
looks to be doubling our draw calls?
//will investigate more shortly
//anim.blendMode = BlendMode.ADD;
anim.alpha = 1.0;
anim.scaleX = attachPointProto.getValue("scale") * 2.0;
anim.scaleY = attachPointProto.getValue("scale") * 1.0;
anim.offsetX = 0;
anim.offsetY = 32;

if
```

```
(debugAttachPoints)
{
if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "Weapon"))
anim.color = 0xFF0000;
else if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "ArcWeapon") ||
StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "DroneBay") ||
StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "SpinalWeapon"))
anim.color = 0xFFFF00;
else if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "Thruster"))
anim.color = 0x00FF00;
else if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "Target"))
anim.color = 0x00FFFF;
else if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "Defense"))
anim.color = 0x0000FF;
}

anim.allowTransform = true;


// Create a sprite for each thruster and add as child of the ship
var thruster:Entity = createEntity();
thruster.add(detail);
thruster.add(anim);

if (debugAttachPoints)
{
anim.alpha = 1.0;
anim.scaleY = attachPointProto.getValue("scale") * 0.1;
anim.scaleX = attachPointProto.getValue("scale") * 0.5;
anim.offsetX = 0;
anim.offsetY = 16;
}

// Add Position component
var position:Position = ObjectPool.get(Position);
thruster.add(position);
position.init(0, 0, 0, LayerEnum.VFX);
BattleUtils.instance.moveToAttachPoint(entity, thruster);

// Add Thruster Component
var thrusterC:Thruster = ObjectPool.get(Thruster);
if(thrusterC != null)
{
if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "ThrusterForward"))
thrusterC.direction = "Forward";
else if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "ThrusterRight"))
thrusterC.direction = "Right";
else if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"),
"ThrusterBackward"))
thrusterC.direction = "Backward";
else
```

```
if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "ThrusterLeft"))
thrusterC.direction = "Left";
else if (debugAttachPoints)
thrusterC.direction = "Backward";
}
thruster.add(thrusterC);

thruster.id = name;
// Add the thruster to the game
_game.addEntity(thruster);
return thruster;
}

public function createMuzzle( entity:Entity, attachPointProto:IPrototype,
weaponProto:IPrototype, slotIndex:Number, visible:Boolean = false ):Entity
{
// Get asset from the weapon proto
var assetName:String = weaponProto.getValue("chargeAsset");
var muzzleAsset:AssetVO = _assetModel.getEntityData(assetName);

// Create a sprite for each muzzle and add as child of the ship
var muzzle:Entity = createEntity();

// Add Detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, muzzleAsset);
detail.prototypeVO = attachPointProto;
muzzle.add(detail);

// Add Animation component
var ownerId:String = Detail(entity.get(Detail)).ownerID;
var ownerFaction:String = _playerModel.getPlayer(ownerId).faction;
var anim:Animation = ObjectPool.get(Animation);
anim.init(muzzleAsset.type, muzzleAsset.spriteName, true, 0, 30, visible);
//PR: Turning the blendmode off on these for now as it causes more of a performance hit and
looks to be doubling our draw calls?
//will investigate more shortly
//anim.blendMode = BlendMode.ADD;
anim.alpha = 1.0;
anim.scaleX = 0.0;
anim.scaleY = 0.0;

// Get color from the weapon data
var color:String = weaponProto.getUnsafeValue("chargeColor");
anim.color = (color) ? uint("0x" + color) : 0xFFFFFF;

anim.allowTransform = true;
muzzle.add(anim);

// Add Position component
var
```

```
position:Position = ObjectPool.get(Position);
muzzle.add(position);
position.init(0, 0, 0, LayerEnum.VFX);
position.rotation = 90.0 * 0.0174532925;
BattleUtils.instance.moveToAttachPoint(entity, muzzle);

// Add Muzzle Component
var muzzleC:Muzzle = ObjectPool.get(Muzzle);
muzzleC.moduleIdx = slotIndex;
muzzleC.currentFrame = -1;
muzzleC.chargeDuration = weaponProto.getUnsafeValue("chargeTime") * 30.0;
muzzleC.weaponClass = weaponProto.getUnsafeValue("itemClass");
muzzleC.baseScale = attachPointProto.getValue("scale");
muzzle.add(muzzleC);

muzzle.id = name;
// Add the thruster to the game
_game.addEntity(muzzle);
return muzzle;
}

public function createDamageEffect( entity:Entity, attachPointProto:IPrototype ):Entity
{
// Create a sprite for each thruster and add as child of the ship
var damageEffect:Entity = ObjectPool.get(Entity);

// Add Detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(CategoryEnum.VFX, _assetModel.getEntityData(TypeEnum.DAMAGE),
attachPointProto);
damageEffect.add(detail);

// Add Animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(Animation(entity.get(Animation)).type, TypeEnum.DAMAGE, true, 0, 30, true);
anim.scaleX = anim.scaleY = attachPointProto.getValue("scale") * (Math.random() * 1.5);
anim.frame = (Math.random() * 18) | 0;
anim.allowTransform = true;
damageEffect.add(anim);

// Add Position component
var position:Position = ObjectPool.get(Position);
damageEffect.add(position);
position.init(0, 0, 0, LayerEnum.VFX);
BattleUtils.instance.moveToAttachPoint(entity, damageEffect);

// Add Damage component
var damage:Damage = ObjectPool.get(Damage);
damage.rotOffset = Math.random() * 360;
damageEffect.add(damage);

damageEffect.id
```

```
= name;
// Add the thruster to the game
_game.addEntity(damageEffect);
return damageEffect
}

public function createDebugLine( data:DebugLineData ):Entity
{
var assetVO:AssetVO = _assetModel.getEntityData(TypeEnum.DEBUG_LINE);
if (!assetVO)
return null;

var debugLine:Entity = createEntity();
// Add Detail component
var detail:Detail = ObjectPool.get(Detail);
detail.init(assetVO.type, assetVO);
debugLine.add(detail);

// Add Animation component
var anim:Animation = ObjectPool.get(Animation);
anim.init(assetVO.type, assetVO.spriteName, false, 0, 30, true);
var xDiff:Number = (data.startX - data.endX) * (data.startX - data.endX);
var yDiff:Number = (data.startY - data.endY) * (data.startY - data.endY);
anim.scaleX = Math.sqrt(xDiff + yDiff) / 64;
anim.scaleY = 0.25;
anim.allowTransform = true;
anim.visible = true;
debugLine.add(anim);

// Add Position component
var position:Position = ObjectPool.get(Position);
position.init(data.startX, data.startY, 0, LayerEnum.ACTIVE_DEFENSE);
position.rotation = Math.atan2(data.endY - data.startY, data.endX - data.startX);
debugLine.add(position);

//debug line component
var dl:DebugLine = ObjectPool.get(DebugLine);
dl.startColor = data.startColor;
dl.endColor = data.endColor;
debugLine.add(dl);

debugLine.id = TypeEnum.DEBUG_LINE + data.id;
// Add the debug line to the game
_game.addEntity(debugLine);
return debugLine;
}

public function destroyAttack( attackIcon:Entity ):void
{
destroyEntity(attackIcon);
ObjectPool.give(attackIcon.remove(Detail));
```

```
ObjectPool.give(attackIcon.remove(Animation));
ObjectPool.give(attackIcon.remove(Grid));
ObjectPool.give(attackIcon.remove(Position));
ObjectPool.give(attackIcon.remove(Attack));
}

public function destroyDebugLine( debugLine:Entity ):void
{
ObjectPool.give(debugLine.remove(Detail));
ObjectPool.give(debugLine.remove(Animation));
ObjectPool.give(debugLine.remove(Position));
ObjectPool.give(debugLine.remove(DebugLine));
ObjectPool.give(debugLine);
}

public function destroyVFX( entity:Entity ):void
{
destroyEntity(entity);
ObjectPool.give(entity.remove(Detail));
ObjectPool.give(entity.remove(Position));
ObjectPool.give(entity.remove(Animation));
if (entity.has(Damage))
ObjectPool.give(entity.remove(Damage));
if (entity.has(Grid))
ObjectPool.give(entity.remove(Grid));
if (entity.has(Thruster))
ObjectPool.give(entity.remove(Thruster));
}

public function destroyTrail( trail:Entity ):void
{
destroyEntity(trail);
ObjectPool.give(trail.remove(Detail));
ObjectPool.give(trail.remove(Position));
ObjectPool.give(trail.remove(Animation));
}

private function get name():String
{
_id++;
return "vfx" + _id;
}

[Inject]
public function set attackFactory( v:IAttackFactory ):void { _attackFactory = v; }
[Inject]
public function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
```

```
public function set viewStack( v:IViewStack ):void { _viewStack = v; }
}
}
```

File 217: igw\com\game\entity\nodes\battle\ActiveDefenseNode.as

```
package com.game.entity.nodes.battle
{
import com.game.entity.components.battle.ActiveDefense;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class ActiveDefenseNode extends Node
{
public var activeDefense:ActiveDefense;
public var animation:Animation;
public var detail:Detail;
public var position:Position;
}
}
```

File 218: igw\com\game\entity\nodes\battle\AreaNode.as

```
package com.game.entity.nodes.battle
{
import com.game.entity.components.battle.Area;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class AreaNode extends Node
{
public var animation:Animation;
public var area:Area;
public var detail:Detail;
public var position:Position;

private var _readyCallback:Function;

public function init( readyCallback:Function ):void
{
_readyCallback = readyCallback;
animation.addListener(onReady);
```

```
}

private function onReady( current:int, animation:Animation ):void
{
if (current == Animation.ANIMATION_READY && _readyCallback != null)
{
animation.removeListener(_readyCallback);
_readyCallback(this);
_readyCallback = null;
}
}

public function destroy():void
{
_readyCallback = null;
animation.removeListener(onReady);
}
}
}

-------------------------------------
File 219: igw\com\game\entity\nodes\battle\BeamNode.as
package com.game.entity.nodes.battle
{
import com.game.entity.components.battle.Beam;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class BeamNode extends Node
{
public var animation:Animation;
public var beam:Beam;
public var detail:Detail;
public var position:Position;

private var _readyCallback:Function;

public function init( readyCallback:Function ):void
{
_readyCallback = readyCallback;
animation.addListener(onReady);
}

private function onReady( current:int, animation:Animation ):void
{
if (current == Animation.ANIMATION_READY && _readyCallback != null)
{
```

```
animation.removeListener(_readyCallback);
_readyCallback(this);
_readyCallback = null;
}
}

public function destroy():void
{
_readyCallback = null;
animation.removeListener(onReady);
}
}
}
```

File 220: igw\com\game\entity\nodes\battle\DebugLineNode.as

```
package com.game.entity.nodes.battle
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.DebugLine;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.render.RenderStarling;

import org.ash.core.Node;

public class DebugLineNode extends Node
{
public var debugLine:DebugLine;
public var position:Position;
public var anim:Animation;
public var render:RenderStarling;
}
}
```

File 221: igw\com\game\entity\nodes\battle\DroneNode.as

```
package com.game.entity.nodes.battle
{
import com.game.entity.components.battle.Drone;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Move;

import org.ash.core.Node;

public class DroneNode extends Node
{
public var animation:Animation;
public
```

```
var drone:Drone;
public var detail:Detail;
public var position:Position;
public var move:Move;
}
}
```

File 222: igw\com\game\entity\nodes\battle\HealthNode.as

```
package com.game.entity.nodes.battle
{
import com.game.entity.components.battle.Health;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.VCList;

import org.ash.core.Node;

public class HealthNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var health:Health;
public var vcList:VCList;

private var _callback:Function;

public function init( callback:Function ):void
{
_callback = callback;
health.addListener(onHealthChanged);
}

private function onHealthChanged( percent:Number, change:Number ):void { _callback &&
_callback(this, percent, change); }

public function destroy():void
{
_callback = null;
health.removeListener(onHealthChanged);
}
}
}
```

File 223: igw\com\game\entity\nodes\battle\ShieldNode.as

```
package com.game.entity.nodes.battle
{
import com.game.entity.components.battle.Shield;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.VCList;
import
```

```
com.game.entity.components.starbase.Building;

import org.ash.core.Node;

public class ShieldNode extends Node
{
public var animation:Animation;
public var $building:Building;
public var shield:Shield;
public var vcList:VCList;

private var _enabledCallback:Function;
private var _strengthCallback:Function;

public function init( enabledCallback:Function, strengthCallback:Function ):void
{
_enabledCallback = enabledCallback;
_strengthCallback = strengthCallback;
shield.addEnableListener(onEnableChanged);
shield.addStrengthListener(onStrengthChanged);
}

private function onStrengthChanged( current:int ):void { _strengthCallback(this, current); }
private function onEnableChanged( enabled:Boolean ):void { _enabledCallback(this, enabled); }

public function destroy():void
{
_enabledCallback = null;
_strengthCallback = null;
shield.removeEnableListener(onEnableChanged);
shield.removeStrengthListener(onStrengthChanged);
animation = null;
$building = null;
shield = null;
vcList = null;
}
}
}

------------------------------------
File 224: igw\com\game\entity\nodes\battle\TrailFXNode.as
package com.game.entity.nodes.battle
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Position;
import com.game.entity.components.battle.TrailFX;

import org.ash.core.Node;

public class TrailFXNode extends Node
{
```

```
public var animation:Animation;
public var position:Position;
public var trail:TrailFX;
}
}



------------------------------------
File 225: igw\com\game\entity\nodes\battle\ship\IShipNode.as
package com.game.entity.nodes.battle.ship
{
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;

public interface IShipNode
{
function get animation():Animation;
function get detail():Detail;
function get health():Health;
function get ientity():Entity;
function get inext():IShipNode;
function get move():Move;
function get position():Position;
function get render():IRender;
function get ship():Ship;
}
}

------------------------------------
File 226: igw\com\game\entity\nodes\battle\ship\ShipNode.as
package com.game.entity.nodes.battle.ship
{
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.shared.render.Render;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import
```

```
org.ash.core.Entity;
import org.ash.core.Node;

public class ShipNode extends Node implements IShipNode
{
public var internal_animation:Animation;
public var internal_detail:Detail;
public var internal_health:Health;
public var internal_move:Move;
public var internal_position:Position;
public var internal_render:Render;
public var internal_ship:Ship;

public function get animation():Animation { return internal_animation; }
public function get detail():Detail { return internal_detail; }
public function get health():Health { return internal_health; }
public function get ientity():Entity { return entity; }
public function get inext():IShipNode { return next; }
public function get move():Move { return internal_move; }
public function get position():Position { return internal_position; }
public function get render():IRender { return internal_render; }
public function get ship():Ship { return internal_ship; }
}
}

-----------------------------------
File 227: igw\com\game\entity\nodes\battle\ship\ShipStarlingNode.as
package com.game.entity.nodes.battle.ship
{
import com.game.entity.components.battle.Health;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.shared.render.RenderStarling;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;
import org.ash.core.Node;

public class ShipStarlingNode extends Node implements IShipNode
{
public var internal_animation:Animation;
public var internal_detail:Detail;
public var internal_health:Health;
public var internal_move:Move;
public var internal_position:Position;
public var internal_render:RenderStarling;
public var internal_ship:Ship;

public
```

```
function get animation():Animation { return internal_animation; }
public function get detail():Detail { return internal_detail; }
public function get health():Health { return internal_health; }
public function get ientity():Entity { return entity; }
public function get inext():IShipNode { return next; }
public function get move():Move { return internal_move; }
public function get position():Position { return internal_position; }
public function get render():IRender { return internal_render; }
public function get ship():Ship { return internal_ship; }
}
}

------------------------------------
File 228: igw\com\game\entity\nodes\sector\MissionNode.as
package com.game.entity.nodes.sector
{
import com.game.entity.components.sector.Mission;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class MissionNode extends Node
{
public var detail:Detail;
public var mission:Mission;
public var position:Position;
}
}
------------------------------------
File 229: igw\com\game\entity\nodes\sector\fleet\FleetNode.as
package com.game.entity.nodes.sector.fleet
{
import com.game.entity.components.shared.render.Render;
import com.game.entity.components.sector.Fleet;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;
import org.ash.core.Node;

public class FleetNode extends Node implements IFleetNode
{
public var internal_animation:Animation;
public var internal_detail:Detail;
public var internal_fleet:Fleet;
public var internal_move:Move;
public
```

```
var internal_position:Position;
public var internal_render:Render;

public function get animation():Animation { return internal_animation; }
public function get detail():Detail { return internal_detail; }
public function get fleet():Fleet { return internal_fleet; }
public function get ientity():Entity { return entity; }
public function get inext():IFleetNode { return next; }
public function get move():Move { return internal_move; }
public function get position():Position { return internal_position; }
public function get render():IRender { return internal_render; }
}
}

-----------------------------------
File 230: igw\com\game\entity\nodes\sector\fleet\FleetStarlingNode.as
package com.game.entity.nodes.sector.fleet
{
import com.game.entity.components.shared.render.RenderStarling;
import com.game.entity.components.sector.Fleet;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;
import org.ash.core.Node;

public class FleetStarlingNode extends Node implements IFleetNode
{
public var internal_animation:Animation;
public var internal_detail:Detail;
public var internal_fleet:Fleet;
public var internal_move:Move;
public var internal_position:Position;
public var internal_render:RenderStarling;

public function get animation():Animation { return internal_animation; }
public function get detail():Detail { return internal_detail; }
public function get fleet():Fleet { return internal_fleet; }
public function get ientity():Entity { return entity; }
public function get inext():IFleetNode { return next; }
public function get move():Move { return internal_move; }
public function get position():Position { return internal_position; }
public function get render():IRender { return internal_render; }
}
}

-----------------------------------
File
```

231: igw\com\game\entity\nodes\sector\fleet\IFleetNode.as
```
package com.game.entity.nodes.sector.fleet
{
import com.game.entity.components.battle.Health;
import com.game.entity.components.sector.Fleet;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;

public interface IFleetNode
{
function get animation():Animation;
function get detail():Detail;
function get fleet():Fleet;
function get ientity():Entity;
function get inext():IFleetNode;
function get move():Move;
function get position():Position;
function get render():IRender;
}
}
```

File 232: igw\com\game\entity\nodes\shared\AnimationNode.as
```
package com.game.entity.nodes.shared
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;

import org.ash.core.Node;

public class AnimationNode extends Node
{
public var detail:Detail;
public var animation:Animation;
}
}
```

File 233: igw\com\game\entity\nodes\shared\EnemyNode.as
```
package com.game.entity.nodes.shared
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.Interactable;
import
```

```
com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class EnemyNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var interactable:Interactable;
public var enemy:Enemy;
public var position:Position;
}
}

------------------------------------
File 234: igw\com\game\entity\nodes\shared\FSMNode.as
package com.game.entity.nodes.shared
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.fsm.FSM;

import org.ash.core.Node;

public class FSMNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var fsm:FSM;
}
}

------------------------------------
File 235: igw\com\game\entity\nodes\shared\MoveNode.as
package com.game.entity.nodes.shared
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class MoveNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var move:Move;
public var position:Position;
}
}
```

File 236: igw\com\game\entity\nodes\shared\OwnedNode.as
```
package com.game.entity.nodes.shared
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class OwnedNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var interactable:Interactable;
public var owned:Owned;
public var position:Position;
}
}
```

File 237: igw\com\game\entity\nodes\shared\RenderNode.as
```
package com.game.entity.nodes.shared
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.Position;

import org.ash.core.Node;

public class RenderNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var position:Position;

private var _render3D:IRender;

public function get render():IRender { return animation.render; }
public function set render( v:IRender ):void { animation.render = v; }

public function get render3D():IRender { return _render3D; }
public function set render3D( v:IRender ):void { _render3D = v; }
}
}
```

File 238: igw\com\game\entity\nodes\shared\grid\GridMoveNode.as

```
package com.game.entity.nodes.shared.grid
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;
import org.ash.core.Node;

public class GridMoveNode extends Node implements IGridNode
{
public var internal_animation:Animation;
public var internal_grid:Grid;
public var internal_move:Move;
public var internal_position:Position;

public function get animation():Animation { return internal_animation; }
public function get ientity():Entity { return entity; }
public function get grid():Grid { return internal_grid; }
public function get move():Move { return internal_move; }
public function get position():Position { return internal_position; }
}
}
```

File 239: igw\com\game\entity\nodes\shared\grid\GridNode.as

```
package com.game.entity.nodes.shared.grid
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;
import org.ash.core.Node;

public class GridNode extends Node implements IGridNode
{
public var internal_animation:Animation;
public var internal_grid:Grid;
public var internal_position:Position;

public function get animation():Animation { return internal_animation; }
public function get ientity():Entity { return entity; }
public function get grid():Grid { return internal_grid; }
public function get position():Position { return internal_position; }
}
}
```

File 240: igw\com\game\entity\nodes\shared\grid\IGridNode.as
```
package com.game.entity.nodes.shared.grid
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Position;

import org.ash.core.Entity;

public interface IGridNode
{
function get animation():Animation;
function get ientity():Entity;
function get grid():Grid;
function get position():Position;
}
}
```

File 241: igw\com\game\entity\nodes\shared\visualComponent\IVCNode.as
```
package com.game.entity.nodes.shared.visualComponent
{
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.VCList;

import org.ash.core.Entity;

public interface IVCNode
{
function get ientity():Entity;
function get render():IRender;
function get vcList():VCList;
}
}
```

File 242: igw\com\game\entity\nodes\shared\visualComponent\IVCSpriteNode.as
```
package com.game.entity.nodes.shared.visualComponent
{
public interface IVCSpriteNode extends IVCNode
{
}
}
```

File 243: igw\com\game\entity\nodes\shared\visualComponent\VCNode.as
```
package com.game.entity.nodes.shared.visualComponent
{
import com.game.entity.components.shared.render.Render;
import
```

```
com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.VCList;

import org.ash.core.Entity;
import org.ash.core.Node;

public class VCNode extends Node implements IVCNode
{
public var internal_render:Render;
public var internal_vcList:VCList;

public function get ientity():Entity { return entity; }
public function get render():IRender { return internal_render; }
public function get vcList():VCList { return internal_vcList; }
}
}
```

File 244: igw\com\game\entity\nodes\shared\visualComponent\VCSpriteNode.as

```
package com.game.entity.nodes.shared.visualComponent
{
import com.game.entity.components.shared.render.RenderSprite;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.VCList;

import org.ash.core.Entity;
import org.ash.core.Node;

public class VCSpriteNode extends Node implements IVCSpriteNode
{
public var internal_render:RenderSprite;
public var internal_vcList:VCList;

public function get ientity():Entity { return entity; }
public function get render():IRender { return internal_render; }
public function get vcList():VCList { return internal_vcList; }
}
}
```

File 245: igw\com\game\entity\nodes\shared\visualComponent\VCSpriteStarlingNode.as

```
package com.game.entity.nodes.shared.visualComponent
{
import com.game.entity.components.shared.render.RenderSpriteStarling;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.VCList;

import org.ash.core.Entity;
import org.ash.core.Node;

public
```

```
class VCSpriteStarlingNode extends Node implements IVCSpriteNode
{
public var internal_render:RenderSpriteStarling;
public var internal_vcList:VCList;

public function get ientity():Entity { return entity; }
public function get render():IRender { return internal_render; }
public function get vcList():VCList { return internal_vcList; }
}
}
```

File 246: igw\com\game\entity\nodes\shared\visualComponent\VCStarlingNode.as

```
package com.game.entity.nodes.shared.visualComponent
{
import com.game.entity.components.shared.render.RenderStarling;
import com.game.entity.components.shared.IRender;
import com.game.entity.components.shared.VCList;

import org.ash.core.Entity;
import org.ash.core.Node;

public class VCStarlingNode extends Node implements IVCNode
{
public var internal_render:RenderStarling;
public var internal_vcList:VCList;

public function get ientity():Entity { return entity; }
public function get render():IRender { return internal_render; }
public function get vcList():VCList { return internal_vcList; }
}
}
```

File 247: igw\com\game\entity\nodes\starbase\BuildingNode.as

```
package com.game.entity.nodes.starbase
{
import com.game.entity.components.battle.Health;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Pylon;
import com.game.entity.components.shared.VCList;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.State;

import org.ash.core.Node;

public class BuildingNode extends Node
{
public
```

```
var animation:Animation;
public var building:Building;
public var detail:Detail;
public var $health:Health;
public var position:Position;
public var $pylon:Pylon;
public var $state:State;
public var $vcList:VCList;

private var _callback:Function;

public function init( callback:Function ):void
{
_callback = callback;
if ($health)
{
$health.addListener(onHealthChanged);
_callback(this, $health.percent, 0);
} else
{
building.buildingVO.addHealthListener(onHealthChanged);
_callback(this, building.buildingVO.currentHealth, 0);
}
}

private function onHealthChanged( percent:Number, change:Number ):void
{
//update the buildingVO if needed the health updates are happening on the health component
if ($health)
building.buildingVO.currentHealth = percent;
_callback(this, percent, change);
}

public function destroy():void
{
_callback = null;
if ($health)
$health.removeListener(onHealthChanged);
else
building.buildingVO.removeHealthListener(onHealthChanged);
$health = null;
$pylon = null;
$state = null;
$vcList = null;
}
}
}

------------------------------------
File 248: igw\com\game\entity\nodes\starbase\PlatformNode.as
package
```

```
com.game.entity.nodes.starbase
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.components.starbase.Platform;

import org.ash.core.Node;

public class PlatformNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var platform:Platform;
public var position:Position;
}
}

-------------------------------------
File 249: igw\com\game\entity\nodes\starbase\StateNode.as
package com.game.entity.nodes.starbase
{
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.starbase.State;

import org.ash.core.Node;

public class StateNode extends Node
{
public var animation:Animation;
public var detail:Detail;
public var state:State;
}
}

-------------------------------------
File 250: igw\com\game\entity\systems\battle\ActiveDefenseSystem.as
package com.game.entity.systems.battle
{
import com.game.entity.components.battle.ActiveDefense;
import com.game.entity.factory.IAttackFactory;
import com.game.entity.nodes.battle.ActiveDefenseNode;
import com.util.BattleUtils;

import flash.geom.Point;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import
```

```
org.ash.core.System;

public class ActiveDefenseSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.ActiveDefenseNode")]
public var nodes:NodeList;

private var _attackFactory:IAttackFactory;
private var _game:Game;
private var _point:Point;
private var _sourceLoc:Point;
private var _targetLoc:Point;

override public function addToGame( game:Game ):void
{
_game = game;
_point = new Point();
_sourceLoc = new Point();
_targetLoc = new Point();
}

override public function update( time:Number ):void
{
if (time == 0)
return;

var activeDefense:ActiveDefense;
var beamLength:Number;
var node:ActiveDefenseNode;
var ratio:Number;
for (node = nodes.head; node; node = node.next)
{
activeDefense = node.activeDefense;

//skip if the render is not ready
if (!node.animation.render)
continue;

//if the owner no longer exists. destroy the entity
if (activeDefense.owner.id != activeDefense.ownerID)
{
removeActiveDefense(node.entity);
continue;
}

// Update the current frame
activeDefense.animationTime += time;

ratio = activeDefense.animationTime / activeDefense.animationLength;

if
```

```
(activeDefense.type == ActiveDefense.BEAM)
{
if (!activeDefense.ready)
{
node.animation.offsetY = node.animation.height * 0.5;
activeDefense.ready = true;
}

if (activeDefense.strength < 1)
{
activeDefense.strength += activeDefense.growSpeed;
if (activeDefense.strength > 1)
activeDefense.strength = 1;
}

// Update the point of origin with the moving firer
BattleUtils.instance.getAttachPointLocation(activeDefense.owner,
activeDefense.sourceAttachPoint, _sourceLoc);
node.position.x = _sourceLoc.x;
node.position.y = _sourceLoc.y;

//the target location of where the attack was shot down
_targetLoc.x = activeDefense.hitLocationX;
_targetLoc.y = activeDefense.hitLocationY;

// Stop the beam at the target it's hitting if it hit successfully
_point.setTo(_sourceLoc.x - _targetLoc.x, _sourceLoc.y - _targetLoc.y);
beamLength = _point.length;

// Set the properties to the beam
node.animation.scaleX = beamLength / activeDefense.baseWidth * activeDefense.strength;
node.position.rotation = Math.atan2(_targetLoc.y - _sourceLoc.y, _targetLoc.x - _sourceLoc.x);
} else
{
// Update sprite scale
node.animation.scaleX = node.animation.scaleY = activeDefense.scaleStart +
activeDefense.scaleDelta * ratio;
}

// Update sprite alpha
node.animation.alpha = activeDefense.alphaStart + activeDefense.alphaDelta * ratio;

//destroy the activeDefense interceptor if the animation has reached its' end
if (activeDefense.animationTime > activeDefense.animationLength)
removeActiveDefense(node.entity);

node.position.dirty = true;
}
}

private
```

```
function removeActiveDefense( entity:Entity ):void
{
_attackFactory.destroyAttack(entity);
}

override public function removeFromGame( game:Game ):void
{
nodes = null;
_game = null;
_point = null;
_sourceLoc = null;
_targetLoc = null;
}

[Inject]
public function set attackFactory( a:IAttackFactory ):void { _attackFactory = a }
}
}

-------------------------------------
File 251: igw\com\game\entity\systems\battle\AreaSystem.as
package com.game.entity.systems.battle
{
import com.game.entity.components.battle.Area;
import com.game.entity.nodes.battle.AreaNode;
import com.util.BattleUtils;

import flash.geom.Point;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class AreaSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.AreaNode")]
public var nodes:NodeList;

private var _game:Game;
private var _sourceLoc:Point;

override public function addToGame( game:Game ):void
{
_game = game;
_sourceLoc = new Point();
nodes.nodeAdded.add(onNodeAdded);
nodes.nodeRemoved.add(onNodeRemoved);
}

private
```

```
function onNodeAdded( node:AreaNode ):void
{
if (node.area.useBeamDynamics)
node.init(onReady);
}

private function onReady( node:AreaNode ):void
{
node.animation.offsetY = node.animation.height * 0.5;
}

private function onNodeRemoved( node:AreaNode ):void
{
node.destroy();
}

override public function update( time:Number ):void
{
var area:Area;
var owner:Entity;
var ratio:Number;

for (var node:AreaNode = nodes.head; node; node = node.next)
{
// Init some useful stuff for updates in general
area = node.area;
owner = _game.getEntity(node.detail.ownerID);

// Perform attachments if the area is attached
if (owner)
{
// If attached to an entity then update location with it
if (area.moveWithSource)
{
BattleUtils.instance.getAttachPointLocation(owner, area.sourceAttachPoint, _sourceLoc);
node.position.x = _sourceLoc.x;
node.position.y = _sourceLoc.y;
}

// If locked to an entity's heading then rotate with it
if (area.rotateWithSource)
{
node.position.rotation = BattleUtils.instance.getAttachPointRotation(owner,
area.sourceAttachPoint);
}
}

// Perform animations if there are any
if (area.animTime < area.animLength)
{
//
```

```
Update the current frame
area.animTime += time;

if (area.animTime > area.animLength)
area.animTime = area.animLength;

ratio = area.animTime / area.animLength;
// Scale the x as required, used for most areas
node.animation.scaleX = area.startScaleX + area.scaleDeltaX * ratio;

// Scale the y as required, usually not for beams
node.animation.scaleY = area.startScaleY + area.scaleDeltaY * ratio;

// Update sprite alpha
node.animation.alpha = area.startAlpha + area.alphaDelta * ratio;
}
/*
else if (node.area.duration > 0)
node.area.resetAnimation();
*/

// Dirty the area so it gets updated
node.position.dirty = true;
}
}

override public function removeFromGame( game:Game ):void
{
nodes.nodeAdded.remove(onNodeAdded);
nodes = null;
_game = null;
_sourceLoc = null;
}
}
}

-------------------------------------
File 252: igw\com\game\entity\systems\battle\BeamSystem.as
package com.game.entity.systems.battle
{
import com.game.entity.components.battle.Beam;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IAttackFactory;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.battle.BeamNode;
import com.util.BattleUtils;

import flash.geom.Point;

import org.ash.core.Entity;
import
```

```
org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class BeamSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.BeamNode")]
public var nodes:NodeList;

private var _game:Game;
private var _point:Point;
private var _sourceLoc:Point;
private var _targetLoc:Point;

private var _vfxFactory:IVFXFactory;
private var _attackFactory:IAttackFactory;

override public function addToGame( game:Game ):void
{
_game = game;
_point = new Point();
_sourceLoc = new Point();
_targetLoc = new Point();
nodes.nodeAdded.add(onNodeAdded);
nodes.nodeRemoved.add(onNodeRemoved);
}

private function onNodeAdded( node:BeamNode ):void
{
node.init(onReady);
}

private function onReady( node:BeamNode ):void
{
node.animation.offsetY = node.animation.height * 0.5;
}

private function onNodeRemoved( node:BeamNode ):void
{
node.destroy();
}

override public function update( time:Number ):void
{
if (time == 0)
return;

var beam:Beam;
var beamLength:Number;
var node:BeamNode;
var
```

```
owner:Entity;
var pos:Position;
var target:Entity;

for (node = nodes.head; node; node = node.next)
{
beam = node.beam;
if (beam.strength < 1)
{
beam.strength += beam.growSpeed;
if (beam.strength > 1)
beam.strength = 1;
}
if (node.animation.render)
{
owner = _game.getEntity(node.detail.ownerID);
// It is possible that the owner of the beam was destroyed before the beam has expired.
// In this case we let the beam own itself so that it continues to exist until the server removes it.
if (!owner)
owner = node.entity;

// Update the point of origin with the moving firer
BattleUtils.instance.getAttachPointLocation(owner, beam.sourceAttachPoint, _sourceLoc);
node.position.x = _sourceLoc.x;
node.position.y = _sourceLoc.y;

target = _game.getEntity(beam.targetID);
if (target && beam.targetID == beam.hitTarget)
{
// Use the specified attach point is there is one
if (beam.targetAttachPoint != "HULL")
{
BattleUtils.instance.getAttachPointLocation(target, beam.targetAttachPoint, _targetLoc);
}
// Use the ship's hull coordinates otherwise
else
{
pos = target.get(Position);
_targetLoc.x = pos.x;
_targetLoc.y = pos.y;
}
} else
{
//used by active defense beams that shoot down projectiles or guided bombs
_targetLoc.x = beam.hitLocationX;
_targetLoc.y = beam.hitLocationY;
}

// Apply the randomization to the location
_targetLoc.x += beam.targetScatterX;
_targetLoc.y
```

```
+= beam.targetScatterY;

// Stop the beam at the target it's hitting if it hit successfully
beamLength = beam.maxRange;
if (beam.attackHit)
{
_point.setTo(_sourceLoc.x - _targetLoc.x, _sourceLoc.y - _targetLoc.y);
beamLength = _point.length;

if (beam.visibleHitCounter <= 0 && beam.strength >= 1)
{
_vfxFactory.createHit(target, node.entity, _targetLoc.x, _targetLoc.y);
beam.visibleHitCounter = 5;
}
}

// Set the properties to the beam
node.animation.scaleX = beamLength / beam.baseWidth * node.beam.strength;
node.position.rotation = Math.atan2(_targetLoc.y - _sourceLoc.y, _targetLoc.x - _sourceLoc.x);

/*
node.animation.scaleX = beam.maxRange / 256 * node.beam.strength;
if (beam.followShipRotation)
{
node.position.rotation = Position(owner.get(Position)).rotation;
} else
node.position.rotation = BattleUtils.instance.getAttachPointRotation(owner,
beam.sourceAttachPoint);
*/
}
}
}

override public function removeFromGame( game:Game ):void
{
nodes.nodeAdded.remove(onNodeAdded);
nodes = null;
_game = null;
_point = null;
_sourceLoc = null;
_targetLoc = null;
}

[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v }
[Inject]
public function set attackFactory( a:IAttackFactory ):void { _attackFactory = a }
}
}

-----------------------------------
```

File 253: igw\com\game\entity\systems\battle\DebugLineSystem.as

```
package com.game.entity.systems.battle
{
import com.enum.TypeEnum;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.battle.DebugLineNode;
import com.model.asset.AssetModel;
import com.service.server.incoming.data.DebugLineData;
import com.service.server.incoming.data.RemovedObjectData;
import com.util.InteractEntityUtil;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class DebugLineSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.DebugLineNode")]
public var nodes:NodeList;

private const UPDATE_INTERVAL_TICKS:int = 4;

private var _assetModel:AssetModel;
private var _game:Game;
private var _lastUpdateTick:int;
private var _vfxFactory:IVFXFactory;

override public function addToGame( game:Game ):void
{
_game = game;
nodes.nodeAdded.add(onNodeAdded);

//build the debug line asset data if it does not exist
if (_assetModel.getEntityData(TypeEnum.DEBUG_LINE) == null)
{
_assetModel.addGameAssetData(InteractEntityUtil.createPrototype(TypeEnum.DEBUG_LINE,
1, "DebugLine"));
}
}

public function addLine( lines:Vector.<DebugLineData> ):void
{
for (var i:int = 0; i < lines.length; i++)
{
_vfxFactory.createDebugLine(lines[i]);
}
}

public
```

```
function removeLine( lines:Vector.<RemovedObjectData> ):void
{
var entity:Entity;
for (var i:int = 0; i < lines.length; i++)
{
entity = _game.getEntity(lines[i].id);
if (entity)
_vfxFactory.destroyDebugLine(entity);
}
}

protected function matchPosition( entityId:String, position:Position ):void
{
var followEntity:Entity = _game.getEntity(entityId);

if (followEntity)
{
var followPosition:Position = followEntity.get(Position);
if (followPosition)
{
position.x = followPosition.x;
position.y = followPosition.y;
}
}
}

protected function onNodeAdded( node:DebugLineNode ):void
{
node.render.setVertexColor(0, node.debugLine.startColor);
node.render.setVertexColor(1, node.debugLine.startColor);
node.render.setVertexColor(2, node.debugLine.endColor);
node.render.setVertexColor(3, node.debugLine.endColor);
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }

public override function removeFromGame( game:Game ):void
{
nodes.nodeAdded.remove(onNodeAdded);
nodes = null;
_assetModel = null;
_game = null;
_vfxFactory = null;
}
}
}
```

File 254: igw\com\game\entity\systems\battle\DroneSystem.as

```
package com.game.entity.systems.battle
{
import com.controller.ServerController;
import com.enum.CategoryEnum;
import com.game.entity.components.battle.Drone;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IAttackFactory;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.battle.DroneNode;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.BeamAttackData;
import com.service.server.incoming.data.ProjectileAttackData;

import flash.geom.Point;
import flash.utils.Dictionary;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class DroneSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.DroneNode")]
public var nodes:NodeList;
[Inject]
public var attackFactory:IAttackFactory;
[Inject]
public var prototypeModel:PrototypeModel;
[Inject]
public var vfxFactory:IVFXFactory;

private var _beamData:BeamAttackData;
private var _game:Game;
private var _id:int;
private var _lookup:Dictionary;
private var _projectileData:ProjectileAttackData;
private var _targetLoc:Point;
private var _targetVec:Point;

override public function addToGame( game:Game ):void
{
_game = game;
_id = 1;
_targetVec
```

```
= new Point();
nodes.nodeRemoved.add(onNodeRemoved);

// Create a default BeamAttackData object for the drones to use
_beamData = new BeamAttackData();
_beamData.attackHit = false;
_beamData.targetAttachPoint = "HULL";
_beamData.targetScatterX = 0;
_beamData.targetScatterY = 0;
_beamData.hitLocation = new Point();

// Create a default ProjectileAttakData object for the drones to use
_projectileData = new ProjectileAttackData();
}

private function onNodeRemoved( node:DroneNode ):void
{
vfxFactory.createExplosion(node.entity, node.position.x, node.position.y);
removeAttack(node.drone);
}

override public function update( time:Number ):void
{
if (time == 0)
return;

var drone:Drone;
var node:DroneNode;
var projectile:Entity;
var target:Entity;

for (node = nodes.head; node; node = node.next)
{
drone = node.drone;

// Early out if the drone is not visible
if (!node.animation.render)
continue;

// Get the target of the drone
target = _game.getEntity(drone.targetID);

// Early out if no target
if (!target)
continue;

// Get the location of the target
_targetLoc = Position(target.get(Position)).position;

// If it's time to clean up the attack then do so
if
```

```
(drone.currentTick >= drone.cleanupTick && drone.weaponAttack)
{
vfxFactory.createHit(target, drone.weaponAttack, _targetLoc.x, _targetLoc.y);
removeAttack(drone);
continue;
}

// Early out if not orbiting anything
if (!drone.isOribiting)
{
// The drone is no longer orbiting so if it was attacking a target, remove that attack.
if (drone.weaponAttack != null)
removeAttack(drone);
continue;
}

// Set the beam origin and target points and compute vector
_targetVec.setTo(_targetLoc.x - node.position.x, _targetLoc.y - node.position.y);

// Orient the drone to point at its target
node.position.rotation = Math.atan2(_targetVec.y, _targetVec.x);

// Advance the time counter
drone.currentTick++;

// If its time to fire at the target then fire
if (drone.currentTick >= drone.nextFireTick)
{
// Set the new next fire tick
drone.nextFireTick += drone.minWeaponTime + (Math.random() * (drone.maxWeaponTime -
drone.minWeaponTime));
drone.cleanupTick = drone.currentTick + (drone.fireDuration * 30.0);

var attackProto:IPrototype = prototypeModel.getWeaponPrototype(drone.weaponProto);
if (attackProto.getValue("attackMethod") == 1)
{
var assetVO:AssetVO = Detail(target.get(Detail)).assetVO;

// Create a dummy set of beam data for the attack
_beamData.attackId = name;
_beamData.entityOwnerId = node.entity.id;
_beamData.targetEntityId = drone.targetID;
_beamData.hitTarget = drone.targetID;
_beamData.start = new Point(node.position.x, node.position.y);
_beamData.maxRange = _targetVec.length - (assetVO.radius * 0.15 * Math.random());
_beamData.weaponPrototype = drone.weaponProto;


// Create the attack
drone.weaponAttack = attackFactory.createBeam(_beamData);
}
```

```
else
{
// Create a dummy set of projectile data for the attack
_projectileData.attackId = name;
_projectileData.entityOwnerId = node.entity.id;
_projectileData.start = new Point(node.position.x, node.position.y);
_projectileData.weaponPrototype = drone.weaponProto;
_projectileData.rotation = node.position.rotation;
_projectileData.guided = false;
_projectileData.fadeTime = 0;
_projectileData.startTick = ServerController.SIMULATED_TICK;
_projectileData.finishTick = ServerController.SIMULATED_TICK + (drone.fireDuration * 20.0);
_projectileData.end.x = _targetLoc.x;
_projectileData.end.y = _targetLoc.y;

// Create the projectile and let it destroy itself when it reaches its target
projectile = attackFactory.createProjectile(null, _projectileData);
Move(projectile.get(Move)).destroyOnComplete = true;
}
}
}
}

private function removeAttack( drone:Drone ):void
{
if (drone.weaponAttack)
{
//when a drone is destroyed at the end of a battle we also need to destroy its' beam
//but that beam may have already been destroyed as part of the cleanup process
//checking for the Detail will tell us if it has already been destroyed
var detail:Detail = drone.weaponAttack.get(Detail);
if (detail && detail.category == CategoryEnum.ATTACK)
attackFactory.destroyAttack(drone.weaponAttack);

drone.weaponAttack = null;
}
}

private function get name():String { _id++; return "DroneBeam" + _id; }

override public function removeFromGame( game:Game ):void
{
nodes.nodeRemoved.remove(onNodeRemoved);
nodes = null;
_beamData = null;
_game = null;
_lookup = null;
_projectileData = null;
_targetLoc = null;
_targetVec = null;
}
```

```
}
}

-----------------------------------
File 255: igw\com\game\entity\systems\battle\ShipSystem.as
package com.game.entity.systems.battle
{
import com.controller.ServerController;
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.game.entity.components.battle.Damage;
import com.game.entity.components.battle.Modules;
import com.game.entity.components.battle.Ship;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Muzzle;
import com.game.entity.components.shared.Position;
import com.game.entity.components.shared.Thruster;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.battle.ship.IShipNode;
import com.game.entity.nodes.battle.ship.ShipNode;
import com.game.entity.nodes.battle.ship.ShipStarlingNode;
import com.model.asset.AssetModel;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.util.BattleUtils;

import flash.geom.Point;
import flash.utils.Dictionary;

import org.adobe.utils.StringUtil;
import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;
import org.console.Console;

public class ShipSystem extends System
{
[Inject(nodeType="IShipNode")]
public var nodes:NodeList;

private var _accelerationComponents:Point;
private var _assetModel:AssetModel;
private var _game:Game;
private var _motionComponentsA:Point;
private var _motionComponentsB:Point
private var _prototypeModel:PrototypeModel;
private var _tempRotation:Number;
private
```

```
var _vfxFactory:IVFXFactory;
private var _soundController:SoundController;

// TEST ONLY - Set TRUE to show colored markers on all attach points
private var _debugAttachPoints:Boolean = false;

override public function addToGame( game:Game ):void
{
var shipNode:ShipNode;
var shipStarlingNode:ShipStarlingNode;
nodes.nodeAdded.add(onNodeAdded);
nodes.nodeRemoved.add(onNodeRemoved);

_accelerationComponents = new Point();
_game = game;
_motionComponentsA = new Point();
_motionComponentsB = new Point();
}

override public function update( time:Number ):void
{
var ship:Ship;
for (var node:IShipNode = nodes.head; node; node = node.inext)
{
ship = node.ship;
// If we need to, set which thruster banks should be visible
// Update every ten frames to reduce flickering from minor changes
if (ServerController.SIMULATED_TICK > ship.lastUpdate + 2)
{
// Set last update
ship.lastUpdate = ServerController.SIMULATED_TICK;

// Update the system's motion history
var p:Point = ship.position1;
p.setTo(node.move.lerpDestination.x, node.move.lerpDestination.y);
ship.position1 = ship.position2;
ship.position2 = ship.position3;
ship.position3 = p;

// Reconstruct the acceleration vector
_motionComponentsA.setTo(ship.position1.x - ship.position2.x,
ship.position1.y - ship.position2.y);
_motionComponentsB.setTo(ship.position2.x - ship.position3.x,
ship.position2.y - ship.position3.y);
_accelerationComponents.setTo(_motionComponentsA.x - _motionComponentsB.x,
_motionComponentsA.y - _motionComponentsB.y);

// Determine the acceleration direction
var accelerationDirection:Number = (Math.atan2(_accelerationComponents.y,
_accelerationComponents.x * -1) + node.move.rotation) * 57.2957795;

//
```

```
Update thruster visibility
ship.thrustersFront = false || _debugAttachPoints;
ship.thrustersRight = false || _debugAttachPoints;
ship.thrustersBack = false || _debugAttachPoints;
ship.thrustersLeft = false || _debugAttachPoints;
if (_accelerationComponents.length > 0.1)
{
/*
if (accelerationDirection > 270 || accelerationDirection < 90)
ship.thrustersFront = true;
if (accelerationDirection > 195 && accelerationDirection < 345)
ship.thrustersRight = true;
*/
if (accelerationDirection > 90 && accelerationDirection < 270)
ship.thrustersBack = true;
/*
if (accelerationDirection > 15 && accelerationDirection < 165)
ship.thrustersLeft = true;
*/
}
}


//thrusters
if (node.move.moving)
{
// Determine sprite frame to use based on angle
_tempRotation = Math.atan2(Math.sin(node.position.rotation), Math.cos(node.position.rotation));
_tempRotation = (_tempRotation / Math.PI) * 180;
_tempRotation = _tempRotation % 360;
if (_tempRotation < 0)
_tempRotation += 360;
node.animation.label = node.detail.spriteName + "_" + Math.round(_tempRotation / 3.025);
}

// Update attachments
for each (var attachment:Entity in ship.attachments)
{
var animation:Animation = attachment.get(Animation);
if (animation)
{
// Update thruster visibility
var thrusterComponent:Thruster = attachment.get(Thruster);
if (thrusterComponent && node.move.moving)
{
if ((ship.thrustersFront && thrusterComponent.direction == "Forward") ||
(ship.thrustersRight && thrusterComponent.direction == "Right") ||
(ship.thrustersBack && thrusterComponent.direction == "Backward") ||
(ship.thrustersLeft && thrusterComponent.direction == "Left"))
{
BattleUtils.instance.moveToAttachPoint(node.ientity,
```

```
attachment, true);
animation.visible = true;
} else
{
animation.visible = false;
}
}

// Update muzzle flash visibility
else
{
// Update the position of the attachment
BattleUtils.instance.moveToAttachPoint(node.ientity, attachment, false);

// Make sure we'll have all the data we need
var modulesComponent:Modules = node.ientity.get(Modules);
var muzzleComponent:Muzzle = attachment.get(Muzzle);
if (modulesComponent && muzzleComponent)
{
// Set the state only if there's data
var moduleStates:Dictionary = modulesComponent.moduleStates;
var moduleIndex:Number = muzzleComponent.moduleIdx;
if ( moduleStates.hasOwnProperty(moduleIndex) )
{
// Animate dynamics as needed
if (muzzleComponent.currentFrame >= 0)
{
// Start playing chargeup sounds on first frame
if (muzzleComponent.currentFrame == 0)
{
if (muzzleComponent.weaponClass == "GravitonPulseNode")
_soundController.playSound(AudioEnum.AFX_GRAVITON_PULSE_NODE_CHARGE, 0.5);
else if (muzzleComponent.weaponClass == "FusionBeamer")
_soundController.playSound(AudioEnum.AFX_FUSION_BEAMER_CHARGE, 0.5);
}

// Increment or decrement the animation frame
var ratio:Number = 0.0;
var scale:Number = 1.0;
if (muzzleComponent.charging)
{
muzzleComponent.currentFrame += 1;
ratio = muzzleComponent.currentFrame / muzzleComponent.chargeDuration;
scale = Math.sin(ratio * 1.57079633);
}
else
{
muzzleComponent.currentFrame -= 4;
ratio = muzzleComponent.currentFrame / muzzleComponent.chargeDuration;
scale = 1.0 - Math.cos(ratio * 1.57079633);
}
```

```
// Clamp and apply scale to sprite
scale = Math.max( 0.0, Math.min( scale, 1.0 ) );
var animComponent:Animation = attachment.get(Animation);
animComponent.scaleX = scale * muzzleComponent.baseScale;
animComponent.scaleY = scale * muzzleComponent.baseScale;
}
// Rest when down spinning down
else if (animation.visible)
{
animation.visible = false;
muzzleComponent.currentFrame = -1;
muzzleComponent.charging = false;
}

// Start charging when entering state
if ( moduleStates[moduleIndex] == 1 )
{
animation.visible = true;
if (!muzzleComponent.charging)
{
muzzleComponent.currentFrame = 0;
muzzleComponent.charging = true;
}
}
// Ramp down during firing
else if (muzzleComponent.charging && moduleStates[moduleIndex] != 2)
{
muzzleComponent.charging = false;
}

}
}
}
}
}

//damage effects
if (node.health.percent > 0 && node.health.percent < node.health.damageThreshold)
{
if (node.ship.damageEffects.length == 0)
showDamage(node);

for each (var damageEffect:Entity in ship.damageEffects)
{
// Update effect position
BattleUtils.instance.moveToAttachPoint(node.ientity, damageEffect);
var position:Position = damageEffect.get(Position);
var damage:Damage = damageEffect.get(Damage);
position.rotation
```

```
+= damage.rotOffset;
}
}
}
}

private function onNodeAdded( node:IShipNode ):void
{
// Set the initial rotation of the ship
var rot:Number = (node.position.rotation / Math.PI) * 180;
rot = rot % 360;
if (rot < 0)
rot += 360;
var num:int = rot / 3.025 | 0;
node.animation.label = node.detail.spriteName + "_" + num;

// Show the thrusters
var apDetail:Detail = node.detail;
var attachPoints:Array = apDetail.prototypeVO.getValue("attachPoints");

// Set the thruster's acceleration threshold
node.ship.accelThreshold = 0.1;
var threshold:Number = apDetail.prototypeVO.getValue("thrusterThreshold");
if (threshold)
node.ship.accelThreshold = threshold;

for each (var attachPoint:String in attachPoints)
{
var attachPointProto:IPrototype = _prototypeModel.getAttachPoint(attachPoint);
var attachPointType:String = attachPointProto.getValue("attachPointType");
var moduleProto:IPrototype = getModuleByAttachPoint(node, attachPoint);
var slotIndex:Number = getModuleIndexByAttachPoint(node, attachPoint);

// Create the proper kind of sprite
if (_debugAttachPoints)
{
var thruster:Entity = _vfxFactory.createThruster(node.ientity, attachPointProto, true);
if(thruster != null)
node.ship.attachments.push(thruster);
}
else if (StringUtil.beginsWith(attachPointType, "ThrusterBackward"))
{
var thruster:Entity = _vfxFactory.createThruster(node.ientity, attachPointProto, false);
if(thruster != null)
node.ship.attachments.push(thruster);
}
else if (StringUtil.beginsWith(attachPointType, "ArcWeapon") && moduleProto &&
moduleProto.getUnsafeValue("chargeTime"))
{
node.ship.attachments.push(_vfxFactory.createMuzzle(node.ientity, attachPointProto,
moduleProto,
```

```
slotIndex));
}
else if (StringUtil.beginsWith(attachPointType, "SpinalWeapon") && moduleProto &&
moduleProto.getUnsafeValue("chargeTime"))
{
node.ship.attachments.push(_vfxFactory.createMuzzle(node.ientity, attachPointProto,
moduleProto, slotIndex));
}
else
{
continue;
}

// Safety check error testing
if (node.ship.attachments.length > 11 && !_debugAttachPoints)
throw new Error("Attempted to add too many attachments!");
}
}

private function getModuleByAttachPoint( node:IShipNode, attachPoint:String ):IPrototype
{
var component:Modules = node.ientity.get(Modules);
return component.getModuleByAttachPoint( attachPoint );
}

private function getModuleIndexByAttachPoint( node:IShipNode, attachPoint:String ):Number
{
var component:Modules = node.ientity.get(Modules);
return component.getModuleIndexByAttachPoint( attachPoint );
}

private function showDamage( node:IShipNode ):void
{
var apDetail:Detail = node.ientity.get(Detail);
var attachPoints:Array = apDetail.prototypeVO.getValue("attachPoints");

// Attach damage effects to each target attach point
var attachPoint:String
for (var i:int = 0; i < attachPoints.length; i++)
{
var attachPointProto:IPrototype = _prototypeModel.getAttachPoint(attachPoints[i]);

// Ignore everything but target points
if (!StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "Target"))
continue;

// Store the damage effect in the component's effects list
node.ship.damageEffects.push(_vfxFactory.createDamageEffect(node.ientity,
attachPointProto));
}
}
```

```
private function onNodeRemoved( node:IShipNode ):void
{
var thrusters:Vector.<Entity> = node.ship.attachments;
for (var i:int = 0; i < thrusters.length; i++)
{
var thruster:Entity = thrusters[i];
if (thruster.has(Detail))
_vfxFactory.destroyVFX(thruster);
}

var damageEffects:Vector.<Entity> = node.ship.damageEffects;
for (i = 0; i < damageEffects.length; i++)
{
var damage:Entity = damageEffects[i];
if (damage.has(Detail))
_vfxFactory.destroyVFX(damage);
}
node.ship.destroy(false);
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }
[Inject]
public function set soundController( v:SoundController ):void { _soundController = v; }

override public function removeFromGame( game:Game ):void
{
nodes.nodeAdded.remove(onNodeAdded);
nodes.nodeRemoved.remove(onNodeRemoved);
nodes = null;

_accelerationComponents = null;
_game = null;
_motionComponentsA = null;
_motionComponentsB = null;
}
}
}

-----------------------------------
File 256: igw\com\game\entity\systems\battle\TrailFXSystem.as
package com.game.entity.systems.battle
{
import com.controller.ServerController;
import
```

```
com.enum.LayerEnum;
import com.game.entity.components.battle.TrailFX;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.battle.TrailFXNode;

import flash.geom.Point;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class TrailFXSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.TrailFXNode")]
public var nodes:NodeList;

private var _alpha:Number;
private var _bounds:Point;
private var _game:Game;
private var _lastUpdate:int;
private var _scalar:Number;
private var _vfxFactory:IVFXFactory;

override public function addToGame( game:Game ):void
{
_bounds = new Point(30, 5);
_game = game;
_lastUpdate = ServerController.SIMULATED_TICK;
_scalar = 1 / _bounds.x;
nodes.nodeRemoved.add(onEntityRemoved);
}

/**
* When an entity is removed we want to clean up any segments that may still exist
* @param node The node that was removed
*/
private function onEntityRemoved( node:TrailFXNode ):void
{
for (var i:int = 0; i < node.trail.segments.length; i++)
{
_vfxFactory.destroyTrail(node.trail.segments[i]);
}
node.trail.segments.length = 0;
}

/**
* Called every frame to update any trails
*
```

```
@param delta The amount of time, in milliseconds, since the last update
*/
override public function update( delta:Number ):void
{
var node:TrailFXNode;
if (ServerController.SIMULATED_TICK > _lastUpdate)
{
//upate and create new trails
for (node = nodes.head; node; node = node.next)
{
if (node.animation.ready)
{
updateCurrentSegment(node, true);
extendTrail(node);
}
}
_lastUpdate = ServerController.SIMULATED_TICK;
} else
{
//update trails
for (node = nodes.head; node; node = node.next)
{
if (node.animation.ready)
updateCurrentSegment(node);
}
}
}

/**
* Takes the current segment and scales it to span the distance covered in the last update
* @param node The trail that we're working with
* @param last Set to true if we're on the last update for the 'currentSegment'
*/
private function updateCurrentSegment( node:TrailFXNode, last:Boolean = false ):void
{
if (node.trail.currentSegment)
{
var animation:Animation = node.trail.currentSegment.get(Animation);
var position:Point = node.position.position;
var segPosition:Position = node.trail.currentSegment.get(Position);
animation.scaleX = Point.distance(position, node.trail.lastPosition) * _scalar;
//if we're on the last update for the current segment then update the rotation just to make sure
we line up properly with the next one
if (last)
segPosition.rotation = Math.atan2(position.y - node.trail.lastPosition.y, position.x -
node.trail.lastPosition.x);
else
segPosition.dirty = true;
}
}

/**
```

```
* Adds a new segment to a trail
* @param node The trail that we're working with
*/
private function extendTrail( node:TrailFXNode ):void
{
var position:Point = node.position.position;
var trail:TrailFX = node.trail;

var segment:Entity;
if (trail.segments.length < trail.maxSegments)
{
segment = _vfxFactory.createTrail(trail, position.x, position.y, Math.atan2(position.y -
trail.lastPosition.y, position.x - trail.lastPosition.x));
} else
{
segment = trail.segments.shift();
var pos:Position = segment.get(Position);
pos.init(position.x, position.y, Math.atan2(position.y - trail.lastPosition.y, position.x -
trail.lastPosition.x), LayerEnum.VFX);
}

var animation:Animation = segment.get(Animation);
animation.scaleX = 0;
animation.scaleY = trail.thickness;
animation.offsetY = trail.thickness * .5;

trail.currentSegment = segment;
trail.lastPosition.setTo(position.x, position.y);
trail.segments.push(segment);

//update the alpha of the remaining segments
_alpha = trail.alphaChange;
for (var i:int = 0; i < trail.segments.length; i++)
{
Animation(trail.segments[i].get(Animation)).alpha = _alpha;
_alpha += trail.alphaChange;
}
if (animation.render)
animation.render.alpha = 0;
}

[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }

public override function removeFromGame( game:Game ):void
{
nodes.nodeRemoved.remove(onEntityRemoved);
nodes = null;
_bounds = null;
_game
```

```
= null;
_vfxFactory = null;
}
}
}
```

File 257: igw\com\game\entity\systems\battle\VitalsSystem.as

```
package com.game.entity.systems.battle
{
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.enum.CategoryEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.Health;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.starbase.Building;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.battle.HealthNode;
import com.game.entity.nodes.battle.ShieldNode;

import flash.utils.Dictionary;

import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class VitalsSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.battle.HealthNode")]
public var healthNodes:NodeList;

[Inject(nodeType="com.game.entity.nodes.battle.ShieldNode")]
public var shieldNodes:NodeList;

private var _soundController:SoundController;
private var _tempID:String;
private var _totalHealthByPlayer:Dictionary;
private var _vfxFactory:IVFXFactory;
public var onHealthChanged:Signal;

override public function addToGame( game:Game ):void
{
var node:HealthNode;
var shNode:ShieldNode;
onHealthChanged = new Signal(String, Number);
_totalHealthByPlayer
```

```
= new Dictionary();
healthNodes.nodeAdded.add(onHealthNodeAdded);
healthNodes.nodeRemoved.add(onNodeRemoved);
shieldNodes.nodeAdded.add(onShieldNodeAdded);
shieldNodes.nodeRemoved.add(onNodeRemoved);
}

private function onHealthNodeAdded( node:HealthNode ):void
{
node.init(onHealthChange);
//track the total health of the player's entities
_tempID = node.detail.ownerID;
if (!_totalHealthByPlayer.hasOwnProperty(_tempID))
{
var health:Health = ObjectPool.get(Health);
health.init(0, 0, null, -1);
_totalHealthByPlayer[_tempID] = health;
}
if (node.detail.category != CategoryEnum.BUILDING || node.detail.prototypeVO.itemClass !=
TypeEnum.PYLON)
{
_totalHealthByPlayer[_tempID].maxHealth += node.health.maxHealth;
_totalHealthByPlayer[_tempID].currentHealth += node.health.currentHealth;
onHealthChanged.dispatch(_tempID, _totalHealthByPlayer[_tempID].percent);
}
}

private function onShieldNodeAdded( node:* ):void
{
node.init(onEnableChanged, onStrengthChanged);
}

private function onNodeRemoved( node:* ):void
{
if (node is HealthNode)
{
onHealthChange(node, 0, node.health.currentHealth);
}
node.destroy();
}

private function onHealthChange( node:HealthNode, percent:Number, change:Number ):void
{
if (node.health.animation)
{
node.health.animation.scaleX = percent;
if (node.health.animation.render)
node.health.animation.render.scaleX = percent;
}
if (node.detail.category != CategoryEnum.BUILDING || node.detail.prototypeVO.itemClass !=
TypeEnum.PYLON)
```

```
{
//update the total health of the player's entities
_tempID = Detail(node.entity.get(Detail)).ownerID;
_totalHealthByPlayer[_tempID].currentHealth -= change;
onHealthChanged.dispatch(_tempID, _totalHealthByPlayer[_tempID].percent);
}
}

private function onStrengthChanged( node:ShieldNode, current:int ):void
{
if (node.shield.enabled && !node.shield.isBuildingShield)
{
//_soundController.playSound(AudioEnum.AFX_SHIELD_HIT_SHIP, 0.5);
if (node.shield.animation)
node.shield.animation.playing = true;
}
}

private function onEnableChanged( node:ShieldNode, enabled:Boolean ):void
{
if (enabled)
{
if (node.shield.isBuildingShield)
{
var building:Building = node.$building;
if (building.buildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM)
{
if (!node.vcList.hasComponentType(TypeEnum.TURRET_SHIELD))
node.vcList.addComponentType(TypeEnum.TURRET_SHIELD);
} else
{
if (!node.vcList.hasComponentType(TypeEnum.BUILDING_SHIELD))
node.vcList.addComponentType(TypeEnum.BUILDING_SHIELD);
}
} else if (!node.vcList.hasComponentType(TypeEnum.SHIELD))
node.vcList.addComponentType(TypeEnum.SHIELD);
_soundController.playSound(AudioEnum.AFX_SHIELDS_UP, 0.5);
//_vfxFactory.createMessage(node.entity, TypeEnum.SHIELD_RESTORED);
} else
{
if (node.shield.isBuildingShield)
{
building = node.$building;
if (building.buildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM)
node.vcList.removeComponentType(TypeEnum.TURRET_SHIELD);
else
node.vcList.removeComponentType(TypeEnum.BUILDING_SHIELD);
} else
node.vcList.removeComponentType(TypeEnum.SHIELD);
//_vfxFactory.createMessage(node.entity,
```

```
TypeEnum.SHIELD_DOWN);
_soundController.playSound(AudioEnum.AFX_SHIELDS_DOWN, 0.5);
}
}

public function getTotalHealthByPlayer( id:String ):Number
{
if (_totalHealthByPlayer.hasOwnProperty(id))
return _totalHealthByPlayer[id].percent;
return 1;
}

[Inject]
public function set soundController( v:SoundController ):void { _soundController = v }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v }

override public function removeFromGame( game:Game ):void
{
onHealthChanged.removeAll();
onHealthChanged = null;
healthNodes.nodeAdded.remove(onHealthNodeAdded);
healthNodes.nodeRemoved.remove(onNodeRemoved);
shieldNodes.nodeAdded.remove(onShieldNodeAdded);
shieldNodes.nodeRemoved.remove(onNodeRemoved);
healthNodes = null;
shieldNodes = null;
_totalHealthByPlayer = null;
}
}
}

-------------------------------------
File 258: igw\com\game\entity\systems\interact\BattleInteractSystem.as
package com.game.entity.systems.interact
{
import com.Application;
import com.controller.ChatController;
import com.controller.GameController;
import com.controller.ServerController;
import com.controller.SettingsController;
import com.controller.keyboard.KeyboardKey;
import com.enum.CategoryEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Owned;
import
```

```
com.game.entity.components.shared.Position;
import com.game.entity.components.starbase.Building;
import com.game.entity.factory.IInteractFactory;
import com.game.entity.nodes.shared.EnemyNode;
import com.game.entity.nodes.shared.OwnedNode;
import com.game.entity.systems.interact.controls.BrowserScheme;
import com.game.entity.systems.interact.controls.ControlledEntity;
import com.game.entity.systems.interact.controls.SelectorEntity;
import com.model.battle.BattleModel;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.player.CurrentUser;
import com.presenter.battle.IBattlePresenter;
import com.util.RangeBuilder;

import flash.geom.Point;
import flash.geom.Rectangle;
import flash.utils.getTimer;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class BattleInteractSystem extends InteractSystem
{
[Inject(nodeType="com.game.entity.nodes.shared.OwnedNode")]
public var owned:NodeList;

[Inject(nodeType="com.game.entity.nodes.shared.EnemyNode")]
public var enemy:NodeList;

public var onControlledUpdated:Signal;

private var _battleModel:BattleModel;
private var _chatController:ChatController;
private var _controlled:Vector.<ControlledEntity>;
private var _ctrlKeyDown:Boolean;
private var _drawSelector:Boolean;
private var _fleetModel:FleetModel;
private var _fleetVO:FleetVO;
private var _futureTargetID:String;
private var _futureTargetTime:Number;
private var _game:Game;
private var _gameController:GameController;
private var _i:int;
private var _interactFactory:IInteractFactory;
private var _isDragSelecting:Boolean;
private var _lastClickTime:Number = 0;
private
```

```
var _lastEntityClicked:Entity;
private var _lastGroupSelected:int = KeyboardKey.ONE.keyCode;
private var _lastTypeSelected:int = 100;
private var _loopI:int;
private var _progressFTEOnMove:Boolean;
private var _rangeBuilder:RangeBuilder;
private var _settingsController:SettingsController;
private var _shiftKeyDown:Boolean;
private var _shipSelectionEntity:Entity;
private var _shipSelectionRect:Rectangle = new Rectangle();
private var _shipTypes:Array = [TypeEnum.FIGHTER, TypeEnum.HEAVY_FIGHTER,
TypeEnum.CORVETTE, TypeEnum.DESTROYER, TypeEnum.BATTLESHIP,
TypeEnum.DREADNOUGHT, TypeEnum.TRANSPORT];
private var _tempControlled:ControlledEntity;

override public function addToGame( game:Game ):void
{
super.addToGame(game);
owned.nodeAdded.add(onNodeAdded);
owned.nodeRemoved.add(onNodeRemoved);
enemy.nodeRemoved.add(onEnemyRemoved);
buildRanges();

_ctrlKeyDown = _shiftKeyDown = false;
_game = game;
_controlled = new Vector.<ControlledEntity>;
onControlledUpdated = new Signal(Vector.<ControlledEntity>);
}

public function init():void
{
_controlScheme = new BrowserScheme();
_controlScheme.init(this, _layer, _keyboardController);
//find the players fleet that is in the battle
if (_battleModel.participants.indexOf(CurrentUser.id) > -1)
{
_fleetVO = _fleetModel.getFleetByBattleAddress(_battleModel.battleServerAddress);
controlAllEntities();
}
}

private function onNodeAdded( node:OwnedNode ):void
{
if(_battleModel.isInstancedMission)
_rangeBuilder.buildRangeFromNodeOnly(node);
else
_rangeBuilder.buildRangeFromNode(node);
}

override public function update( time:Number ):void
{
```

```
super.update(time);
if (_controlled.length > 0)
{
_drawSelector = false;
for (_i = 0; _i < _controlled.length; _i++)
{
_tempControlled = _controlled[_i];
if (_tempControlled.destination)
{
if (!Move(_tempControlled.entity.get(Move)).moving)
_tempControlled.destination = null;
}
//if the ship has a target but no selector is shown then we need to draw the selector
if (!_tempControlled.selectorEnemy && enemy.head != null &&
Attack(_tempControlled.entity.get(Attack)).targetID != null)
_drawSelector = true;
}
if (_drawSelector)
showEnemySelector();

if (_futureTargetID)
{
_futureTargetTime += time;
if (_futureTargetTime >= .15)
assignTarget(_futureTargetID);
}
}
}

override protected function onInteraction_mouseMove( x:Number, y:Number ):void
{
if ((_ctrlKeyDown || _shiftKeyDown) && hasDragStarted(x, y))
{
_isDragSelecting = true;

var startPt:Point = updateScenePt(_startDrag.x, _startDrag.y);
var crntPt:Point = updateScenePt(x, y);

_shipSelectionRect.x = crntPt.x > startPt.x ? startPt.x : crntPt.x;
_shipSelectionRect.y = crntPt.y > startPt.y ? startPt.y : crntPt.y;
_shipSelectionRect.width = Math.abs(crntPt.x - startPt.x); //sizePt.x;
_shipSelectionRect.height = Math.abs(crntPt.y - startPt.y); //sizePt.y;
_shipSelectionEntity = _interactFactory.showMultiShipSelection(_shipSelectionEntity,
_shipSelectionRect);
} else
{
_isDragSelecting = false;
super.onInteraction_mouseMove(x, y);
}
}
```

```
override protected function onInteraction_mouseUp( x:Number, y:Number,
isRightMouse:Boolean = false ):void
{
if (!_isDragSelecting)
super.onInteraction_mouseUp(x, y, isRightMouse);
else
{
if (!_shiftKeyDown)
uncontrolAllEntities();

var selectedShips:Vector.<Entity> = Vector.<Entity>([]);
if (owned && owned.head)
{
var node:OwnedNode = owned.head;
var pos:Position;
var type:String;

while (node)
{
type = node.detail.category;
pos = node.position;

if (_shipSelectionRect.contains(pos.x, pos.y))
selectedShips.push(node.entity);

node = node.next;
}
}

if (selectedShips.length > 0)
{
for each (var entity:Entity in selectedShips)
controlEntity(entity);
}

//cleanup
_interactFactory.destroyInteractEntity(_shipSelectionEntity);
_shipSelectionEntity = null;
}

_isDragSelecting = false;

if (isRightMouse)
_isRightMouseDown = false; //for now we're going to say that only one mouse can be down at a
time..
}

override protected function onClick( dx:Number, dy:Number ):Vector.<Entity>
{
```

```
if( _battleModel.isReplay )
{
return null;
}
//see what the player is clicking on
var interacts:Vector.<Entity> = super.onClick(dx, dy);
var move:Move;
var position:Position;
var tempTime:Number = getTimer();

if (interacts.length > 0)
{
for (_loopI = 0; _loopI < interacts.length; _loopI++)
{
if (_controlled.length != 0 && interacts[_loopI].get(Enemy))
{
assignTarget(interacts[_loopI].id, true);
updateControlled();

if (_inFTE)
progressFTE();

break;
} else if (interacts[_loopI].get(Owned))
{
//switch to this entity owned by the player if we cannot find an enemy
if (!interacts[_loopI].has(Building) || Building(interacts[_loopI].get(Building)).buildingVO.itemClass
== TypeEnum.POINT_DEFENSE_PLATFORM)
{
if (interacts[_loopI] == _lastEntityClicked && tempTime - _lastClickTime < 250)
{
if (!_shiftKeyDown)
uncontrolAllEntities();

controlAllEntitiesOfType(interacts[_loopI]);
} else if (Interactable(interacts[_loopI].get(Interactable)).selected && _shiftKeyDown)
{
uncontrolEntity(interacts[_loopI]);
break;
} else if (!Interactable(interacts[_loopI].get(Interactable)).selected || !_shiftKeyDown)
{
//get rid of all the controlled entities except the one that we want if shift is not down
if (!_shiftKeyDown && (_controlled.length != 1 || _controlled[0].entity != interacts[_loopI]))
uncontrolAllEntities();

assignTarget(_futureTargetID, true);
controlEntity(interacts[_loopI]);
_lastEntityClicked = interacts[_loopI];

if
```

```
(tempTime - _lastClickTime < 150)
controlAllEntitiesOfType(interacts[_loopI]);

else
updateControlled();

if (_inFTE)
progressFTE();

break;
}
}
}
}
} else if (_controlled)
{
//player is trying to move their ship
moveControlledToLocation(dx, dy);
}

_lastClickTime = tempTime;

return interacts;
}

override public function onKey( keyCode:uint, up:Boolean = true ):void
{
if (!_fteController.running && (_chatController.chatHasFocus ||
_viewController.modalHasFocus))
return;

var cnt:int = 0;
var direction:int;
var eSelected:Entity;
var num:int;
var temp:Entity;

super.onKey(keyCode, up);
switch (keyCode)
{
//left / right
case KeyboardKey.A.keyCode:
{
if (_ctrlKeyDown)
{
controlAllEntities();
break;
}
}

case
```

```
KeyboardKey.D.keyCode:
case KeyboardKey.LEFT.keyCode:
case KeyboardKey.RIGHT.keyCode:
{
if (_shiftKeyDown && _controlled.length == 6)
return;

eSelected = _controlled.length > 0 ? _controlled[0].entity : null;
direction = (keyCode == KeyboardKey.A.keyCode || keyCode == KeyboardKey.LEFT.keyCode)
? 0 : 1;

var node:OwnedNode;
if (eSelected)
{
for (node = owned.head; node; node = node.next)
{
//find the currently selected node
if (node.entity == eSelected)
break;
}
} else if (owned.head)
node = owned.head;

temp = (eSelected) ? eSelected : (node) ? node.entity : null;

while (node && cnt != 1)
{
if (direction == 0)
node = (node.previous) ? node.previous : owned.tail;
else
node = (node.next) ? node.next : owned.head;

if (node.entity == temp)
cnt++;
else
{
if (!node.interactable.selected || !_shiftKeyDown)
{
if (!_shiftKeyDown)
{
assignTarget(_futureTargetID);
uncontrolAllEntities();
}

controlEntity(node.entity);
break;
}
}
}

if
```

```
(cnt == 1 && node != null && !eSelected && node.detail.category == CategoryEnum.SHIP)
{
if (!node.interactable.selected || !_shiftKeyDown)
{
if (!_shiftKeyDown)
{
assignTarget(_futureTargetID);
uncontrolAllEntities();
}
controlEntity(node.entity);
}
}

updateControlled();
if (_inFTE)
progressFTE();

break;
}

//up / down
case KeyboardKey.DOWN.keyCode:
case KeyboardKey.S.keyCode:
case KeyboardKey.UP.keyCode:
case KeyboardKey.W.keyCode:
{
if (_controlled.length > 0)
{
eSelected = _controlled[0].selectedEnemy;
direction = (keyCode == KeyboardKey.S.keyCode || keyCode ==
KeyboardKey.DOWN.keyCode) ? 0 : 1;

var eNode:EnemyNode;

if (eSelected)
{
for (eNode = enemy.head; eNode; eNode = eNode.next)
{
//find the currently selected node
if (eNode.entity == eSelected)
break;
}
} else if (enemy.head)
eNode = enemy.head;

temp = (eSelected) ? eSelected : (eNode) ? eNode.entity : null;

while (eNode && cnt != 1)
{
if (direction == 0)
eNode
```

```
= (eNode.previous) ? eNode.previous : enemy.tail;
else
eNode = (eNode.next) ? eNode.next : enemy.head;

if (eNode.entity == temp)
cnt++;
else
{
_futureTargetID = eNode.entity.id;
_futureTargetTime = 0;
setEnemy(eNode.entity.id);
break;
}
}

if (cnt == 1 && eNode != null && !eSelected)
{
_futureTargetID = eNode.entity.id;
_futureTargetTime = 0;
setEnemy(eNode.entity.id);
}
showEnemySelector();
}

if (_inFTE)
progressFTE();

break;
}

//shift
case KeyboardKey.CONTROL.keyCode:
{
_ctrlKeyDown = !up;
if (up && _shipSelectionEntity)
{
_startDrag.setTo(Application.STAGE.mouseX, Application.STAGE.mouseY);
_interactFactory.destroyInteractEntity(_shipSelectionEntity);
_shipSelectionEntity = null;
}
break;
}

//control
case KeyboardKey.SHIFT.keyCode:
{
_shiftKeyDown = !up;
if (up && _shipSelectionEntity)
{
_startDrag.setTo(Application.STAGE.mouseX, Application.STAGE.mouseY);
_interactFactory.destroyInteractEntity(_shipSelectionEntity);
```

```
_shipSelectionEntity = null;
}
break;
}

//Q key, select all
case KeyboardKey.Q.keyCode:
{
controlAllEntities();
break;
}

//deselect all entities
case KeyboardKey.ESCAPE.keyCode:
{
uncontrolAllEntities();
break;
}

//function keys
case KeyboardKey.F1.keyCode:
case KeyboardKey.F2.keyCode:
case KeyboardKey.F3.keyCode:
case KeyboardKey.F4.keyCode:
case KeyboardKey.F5.keyCode:
case KeyboardKey.F6.keyCode:
{
if (_fleetVO)
{
var tempTime:Number = getTimer();
num = keyCode - 112;
eSelected = _game.getEntity(_fleetVO.getShipIDByIndex(num));

if (eSelected)
{
if (eSelected == _lastEntityClicked && tempTime - _lastClickTime < 250)
{
if (!_shiftKeyDown)
uncontrolAllEntities();
controlAllEntitiesOfType(eSelected);
} else if (Interactable(eSelected.get(Interactable)).selected && _shiftKeyDown)
{
uncontrolEntity(eSelected);
break;
} else if (!Interactable(eSelected.get(Interactable)).selected || !_shiftKeyDown)
{
//get rid of all the controlled entities except the one that we want if shift is not down
if (!_shiftKeyDown && (_controlled.length != 1 || _controlled[0].entity != eSelected))
uncontrolAllEntities();
assignTarget(_futureTargetID,
```

```
true);
controlEntity(eSelected);
_lastEntityClicked = eSelected;
updateControlled();
_lastClickTime = tempTime;
break;
}
}
}

break;
}

//number keys
case KeyboardKey.ONE.keyCode:
case KeyboardKey.TWO.keyCode:
case KeyboardKey.THREE.keyCode:
case KeyboardKey.FOUR.keyCode:
case KeyboardKey.FIVE.keyCode:
case KeyboardKey.SIX.keyCode:
case KeyboardKey.SEVEN.keyCode:
case KeyboardKey.EIGHT.keyCode:
case KeyboardKey.NINE.keyCode:
case KeyboardKey.ZERO.keyCode:
{
if (_fleetVO)
{
//assign group
if (_ctrlKeyDown)
{
var a:String = '';
for each (var ctrlEntity:ControlledEntity in _controlled)
a += _fleetVO.getShipIndexByID(ctrlEntity.entity.id);
_fleetVO.fleetGroupData[keyCode] = a;
_settingsController.save();
}
//select group
else
{
a = _fleetVO.fleetGroupData[keyCode];
if (a && a.length > 0)
{
uncontrolAllEntities();
var entity:Entity;
var id:String;
var index:int;
for (var idx:int = 0; idx < a.length; idx++)
{
index = int(a.charAt(idx));
id = _fleetVO.getShipIDByIndex(index);
if
```

```
(id)
{
entity = _game.getEntity(id);
if (entity)
controlEntity(entity);
}
}
} else if (keyCode < KeyboardKey.SEVEN.keyCode)
onKey(KeyboardKey.F1.keyCode + (keyCode - KeyboardKey.ONE.keyCode));
}
_lastGroupSelected = keyCode;
}

break;
}

case KeyboardKey.T.keyCode:
case KeyboardKey.R.keyCode:
{
cnt = 0;
var found:Boolean;
var dir:int = 1; //keyCode == KeyboardKey.T.keyCode ? 1 : -1;
var shipType:String = getNextShipType(dir);
while (!found && cnt <= 6)
{
if (checkOwnedForShipType(shipType))
found = true;
shipType = getNextShipType(dir);
cnt++;
}

var detail:Detail;
var selectedShips:Vector.<Entity> = Vector.<Entity>([]);
node = owned.head;
while (node)
{
detail = node.entity.get(Detail);

if (detail)
{
var split:Array = detail.type.split("_");
var trimmed:String = split.length > 0 ? split[0] : "";
if (trimmed && trimmed.toLowerCase() == shipType.toLowerCase())
selectedShips.push(node.entity);
}
node = node.next;
}

if (selectedShips.length > 0)
{
uncontrolAllEntities();
```

```
for each (entity in selectedShips)
controlEntity(entity);
}
break;
}

case KeyboardKey.F.keyCode:
case KeyboardKey.G.keyCode:
{
if (_fleetVO)
{
if (_lastGroupSelected < 0)
break;
cnt = 0;
found = false;
var tmpGroupKeyCode:int = _lastGroupSelected;
var i:int = keyCode == KeyboardKey.F.keyCode ? -1 : 1;
while (!found && cnt <= 10)
{
//advance the numeric group assignment
tmpGroupKeyCode += i;

if (tmpGroupKeyCode < KeyboardKey.ZERO.keyCode)
tmpGroupKeyCode = KeyboardKey.NINE.keyCode;
else if (tmpGroupKeyCode > KeyboardKey.NINE.keyCode)
tmpGroupKeyCode = KeyboardKey.ZERO.keyCode;

if (_fleetVO.fleetGroupData.hasOwnProperty(tmpGroupKeyCode))
{
found = true;
}
cnt++;
}
if (found)
onKey(tmpGroupKeyCode);
}
break;
}
}
}

private function checkOwnedForShipType( type:String ):Boolean
{
var found:Boolean;
var detail:Detail;
var node:OwnedNode = owned.head;

while (!found && node)
{
detail
```

```
= node.entity.get(Detail);

if (detail && detail.type.indexOf(type) > -1)
found = true;

node = node.next;
}

return found;
}

private function getNextShipType( direction:int = 1 ):String
{
var idx:int = _lastTypeSelected + direction;

if (idx < 0)
idx = _shipTypes.length - 1;

if (idx > _shipTypes.length - 1)
idx = 0;

_lastTypeSelected = idx;

return _shipTypes[idx];
}

public function selectOwnedShipByID( id:String ):void
{
//get the index of the id
if (_fleetVO)
{
//call onkey with the keycode of a function key to select the ship
var index:int = _fleetVO.getShipIndexByID(id);
if (index > -1)
onKey(KeyboardKey.F1.keyCode + index, true);
}
}

public function updateControlled():void
{
if (presenter)
{
if (_controlled.length > 0)
{
showDestination();
showEnemySelector();
}
}
}

private
```

```
function showDestination():void
{
var index:int;
var key:String;
var move:Move;

for (_i = 0; _i < _controlled.length; _i++)
{
_tempControlled = _controlled[_i];
if (_tempControlled.entity.has(Building))
continue;
move = _tempControlled.entity.get(Move);

if (!move)
{
if (_tempControlled.destination)
_tempControlled.destination = null;

continue;
}

key = int(move.destination.x) + "-" + int(move.destination.y);

if (SelectorEntity.getSelectorEntity(key))
_tempControlled.destination = SelectorEntity.getSelectorEntity(key);

else
_tempControlled.destination = createSelectorEntity(key, _interactFactory.showSelection(null,
null, move.destination.x, move.destination.y, true));
}
}

private function showEnemySelector():void
{
var attack:Attack;
for (_i = 0; _i < _controlled.length; _i++)
{
_tempControlled = _controlled[_i];
attack = _tempControlled.entity.get(Attack);

if (!attack || attack.targetID == null)
{
_tempControlled.selectorEnemy = null;
_tempControlled.selectedEnemy = null;
continue;
} else
{
_tempControlled.selectedEnemy = _game.getEntity(attack.targetID);

if (!_tempControlled.selectedEnemy)
{
```

```
_tempControlled.selectorEnemy = null;
_tempControlled.selectedEnemy = null;
} else
{
if (SelectorEntity.getSelectorEntity(attack.targetID))
_tempControlled.selectorEnemy = SelectorEntity.getSelectorEntity(attack.targetID);
else
_tempControlled.selectorEnemy = createSelectorEntity(attack.targetID,
_interactFactory.showSelection(_tempControlled.selectedEnemy, null,0,0,true));
}
}
}
}

private function setEnemy( enemyID:String ):void
{
var attack:Attack;

for (_i = 0; _i < _controlled.length; _i++)
{
_tempControlled = _controlled[_i];
attack = _tempControlled.entity.get(Attack);
attack.targetID = enemyID;
}
}

private function controlEntity( entity:Entity ):void
{
if( _battleModel.isReplay )
{
return;
}
var attack:Attack = entity.get(Attack);
var interactable:Interactable = entity.get(Interactable);

if (!interactable.selected)
{
interactable.selected = true;

var controlled:ControlledEntity = ObjectPool.get(ControlledEntity);
controlled.entity = entity;
controlled.selectedEnemy = _game.getEntity(attack.targetID);
controlled.selector = _interactFactory.showSelection(controlled.entity,
controlled.selector,0,0,true);
if (entity.has(Building))
controlled.range = _interactFactory.createRange(controlled.entity);
else
controlled.range = _interactFactory.createShipRange(controlled.entity);
_controlled.push(controlled);
onControlledUpdated.dispatch(_controlled);
```

```
}
}

private function controlAllEntitiesOfType( entity:Entity ):void
{
var added:Boolean = false;
var detail:Detail = entity.get(Detail);

for (var node:OwnedNode = owned.head; node; node = node.next)
{
if (!node.interactable.selected && Detail(node.entity.get(Detail)).type == detail.type)
{
controlEntity(node.entity);
added = true;
}
}

if (added)
updateControlled();
}

private function controlAllEntities():void
{
var added:Boolean = false;

for (var node:OwnedNode = owned.head; node; node = node.next)
{
if (!node.interactable.selected)
{
controlEntity(node.entity);
added = true;
}
}

if (added)
updateControlled();
}

private function uncontrolEntity( entity:Entity ):void
{
for (_i = 0; _i < _controlled.length; _i++)
{
if (_controlled[_i].entity == entity)
{
_tempControlled = _controlled[_i];
if (entity.has(Interactable))
Interactable(entity.get(Interactable)).selected = false;
_tempControlled.selectedEnemy = null;
_tempControlled.destination = null;
_interactFactory.destroyInteractEntity(_tempControlled.range);
```

```
_interactFactory.destroyInteractEntity(_tempControlled.selector);
ObjectPool.give(_tempControlled);
_controlled.splice(_i, 1);
break;
}
}
}

private function uncontrolAllEntities():void
{
for (_i = 0; _i < _controlled.length; _i++)
{
_tempControlled = _controlled[_i];
Interactable(_tempControlled.entity.get(Interactable)).selected = false;
_tempControlled.selectedEnemy = null;
_tempControlled.destination = null;
_interactFactory.destroyInteractEntity(_tempControlled.range);
_interactFactory.destroyInteractEntity(_tempControlled.selector);
ObjectPool.give(_tempControlled);
}

_controlled.length = 0;
}

private function moveControlledToLocation( dx:Number, dy:Number ):void
{
var move:Move;
var position:Position;

for (_i = 0; _i < _controlled.length; _i++)
{
_tempControlled = _controlled[_i];
if (_tempControlled.entity.has(Building))
continue;
move = _tempControlled.entity.get(Move);

if (move)
{
position = Position(_tempControlled.entity.get(Position));
var dest:Point = updateScenePt(dx, dy);
//only send the message to move if the destination is far enough away from their last destination

if (Math.abs(move.destination.x - dest.x) > 10 || Math.abs(move.destination.y - dest.y) > 10)
{
move.setDestination(dest.x, dest.y);
_gameController.battleMoveShip(_tempControlled.entity.id, dest.x, dest.y,
ServerController.SIMULATED_TICK);
}
}
}
```

```
showDestination();
if (_progressFTEOnMove)
{
_progressFTEOnMove = false;
progressFTE();
}
}

private function assignTarget( target:String, moveToTarget:Boolean = false ):void
{
if (_controlled.length > 0 && target)
{
for (_i = 0; _i < _controlled.length; _i++)
{
Attack(_controlled[_i].entity.get(Attack)).targetID = target;
_gameController.battleAttackShip(_controlled[_i].entity.id, target, moveToTarget);
}
}

_futureTargetID = null;
_futureTargetTime = 0;
}

private function onNodeRemoved( node:OwnedNode ):void
{
uncontrolEntity(node.entity);
}

private function onEnemyRemoved( node:EnemyNode ):void
{
for (_i = 0; _i < _controlled.length; _i++)
{
if (_controlled[_i].selectedEnemy == node.entity)
{
_tempControlled = _controlled[_i];
_tempControlled.selectorEnemy = null;
_tempControlled.selectedEnemy = null;
Attack(_tempControlled.entity.get(Attack)).targetID = null;

}
}
}

private function createSelectorEntity( id:String, entity:Entity ):SelectorEntity
{
if (entity == null)
return null;
var selector:SelectorEntity = ObjectPool.get(SelectorEntity);
selector.init(id, entity, _interactFactory);
return
```

```
selector;
}

public function buildRanges():void
{
for (var on:OwnedNode = owned.head; on; on = on.next)
{
if(_battleModel.isInstancedMission)
_rangeBuilder.buildRangeFromNodeOnly(on);
else
_rangeBuilder.buildRangeFromNode(on);
}
}

public function toggleFTEProgressOnMove():void { _progressFTEOnMove = true; }

[Inject]
public function set battleModel( value:BattleModel ):void { _battleModel = value; }
[Inject]
public function set chatController( value:ChatController ):void { _chatController = value; }
[Inject]
public function set fleetModel( value:FleetModel ):void { _fleetModel = value; }
[Inject]
public function set gameController( value:GameController ):void { _gameController = value; }
[Inject]
public function set interactFactory( value:IInteractFactory ):void { _interactFactory = value; }
public function set presenter( v:IBattlePresenter ):void { _presenter = v; }
public function get presenter():IBattlePresenter { return IBattlePresenter(_presenter); }
[Inject]
public function set rangeBuilder( v:RangeBuilder ):void { _rangeBuilder = v; }
[Inject]
public function set settingsController( v:SettingsController ):void { _settingsController = v; }

override public function removeFromGame( game:Game ):void
{
super.removeFromGame(game);
enemy.nodeRemoved.remove(onEnemyRemoved);
enemy = null;
owned.nodeAdded.remove(onNodeAdded);
owned.nodeRemoved.remove(onNodeRemoved);
owned = null;

onControlledUpdated.removeAll();
onControlledUpdated = null;

uncontrolAllEntities();
_chatController = null;
_game = null;
_gameController = null;
_interactFactory.clearRanges();
_interactFactory
```

```
= null;
_rangeBuilder = null;
_controlled = null;
_settingsController = null;
_shipSelectionEntity = null;
_tempControlled = null;
}
}
}

------------------------------------
File 259: igw\com\game\entity\systems\interact\InteractSystem.as
package com.game.entity.systems.interact
{
import com.Application;
import com.controller.fte.FTEController;
import com.controller.keyboard.KeyboardController;
import com.controller.keyboard.KeyboardKey;
import com.enum.CategoryEnum;
import com.event.signal.InteractSignal;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Interactable;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Position;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.Platform;
import com.game.entity.systems.interact.controls.IControlScheme;
import com.game.entity.systems.shared.grid.GridSystem;
import com.model.asset.AssetVO;
import com.model.scene.SceneModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;

import flash.events.Event;
import flash.events.IEventDispatcher;
import flash.events.MouseEvent;
import flash.geom.Point;
import flash.geom.Rectangle;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.System;
import org.parade.core.IViewStack;
import org.parade.core.ViewController;
import org.parade.core.ViewEvent;
import org.parade.enum.ViewEnum;
import org.robotlegs.extensions.presenter.api.IPresenter;

public class InteractSystem extends System
{
```

```
private static const DECAY:Number = .9;
private static const THRESHOLD:int = 5;

protected var _bbox:Rectangle;
protected var _begunInteraction:Boolean;
protected var _controlScheme:IControlScheme;
protected var _dragging:Boolean;
protected var _entityToFollow:Entity;
protected var _eventDispatcher:IEventDispatcher;
protected var _fteController:FTEController;
protected var _gridSystem:GridSystem;
protected var _inFTE:Boolean;
protected var _interactSignal:InteractSignal;
protected var _isRightMouseDown:Boolean;
protected var _keyboardController:KeyboardController;
protected var _layer:*;
protected var _minZoom:Number;
protected var _maxZoom:Number;
protected var _point:Point;
protected var _presenter:IPresenter;
protected var _residualScroll:Boolean;
protected var _sceneModel:SceneModel;
protected var _scrollDelta:Point;
protected var _scrollDeltaStart:Point;
protected var _starbaseModel:StarbaseModel;
protected var _startDrag:Point;
protected var _stopScroll:Boolean;
protected var _targetViewArea:Boolean;
protected var _targetViewAreaDelta:Point;
protected var _targetViewAreaStart:Point;
protected var _targetViewAreaTime:Number;
protected var _targetViewAreaTotalTime:Number;
protected var _tempPoint:Point;
protected var _viewStack:IViewStack;


override public function addToGame( game:Game ):void
{
_begunInteraction = _dragging = _residualScroll = false;
_bbox = new Rectangle();
_gridSystem = GridSystem(game.getSystem(GridSystem));
_inFTE = false;
_layer = _viewStack.getLayer(ViewEnum.GAME);
_minZoom = 0.5;
_maxZoom = 1.2;
_point = new Point();
_scrollDelta = new Point();
_scrollDeltaStart = new Point();
_startDrag = new Point();
_stopScroll
```

```
= false;
_targetViewArea = false;
_targetViewAreaDelta = new Point();
_targetViewAreaStart = new Point();
_tempPoint = new Point();

_eventDispatcher.addEventListener(ViewEvent.SHOW_VIEW, onShowView);
}

override public function update( time:Number ):void
{
//residual scroll
if (_residualScroll)
{
adjustLocation(_scrollDelta.x, _scrollDelta.y);
_scrollDelta.x *= DECAY;
_scrollDelta.y *= DECAY;
if (Math.abs(_scrollDelta.x) < .4 && Math.abs(_scrollDelta.y) < .4)
_residualScroll = false;
} else if (_targetViewArea)
{
_targetViewAreaTime += time;
var ratio:Number = Math.min(_targetViewAreaTotalTime, _targetViewAreaTime) /
_targetViewAreaTotalTime;
jumpToLocation(_targetViewAreaStart.x + ratio * _targetViewAreaDelta.x,
_targetViewAreaStart.y + ratio * _targetViewAreaDelta.y);
if (ratio >= 1)
_targetViewArea = false;
} else if (_entityToFollow)
{
if (_entityToFollow.has(Move))
{
var position:Position = _entityToFollow.get(Position);
jumpToLocation(position.x, position.y);
} else
_entityToFollow = null;
}
}

public function moveToLocation( lx:Number, ly:Number, time:Number = 1 ):void
{
if (_sceneModel.ready)
{
_targetViewArea = true;
_targetViewAreaStart.setTo(_sceneModel.focus.x, _sceneModel.focus.y);
_targetViewAreaDelta.setTo(lx - _targetViewAreaStart.x, ly - _targetViewAreaStart.y);
_targetViewAreaTime = 0;
_targetViewAreaTotalTime = time;
} else
jumpToLocation(lx, ly);
}
```

```
public function jumpToLocation( lx:Number, ly:Number ):void
{
if (_sceneModel.ready)
{
_sceneModel.setFocus(lx, ly);
_interactSignal.scroll(lx, ly);
}
}

public function adjustLocation( lx:Number, ly:Number ):void
{
if (_sceneModel.ready)
{
_sceneModel.adjustFocus(lx, ly);
_interactSignal.scroll(lx, ly);
}
}

public function updateScenePt( dx:Number, dy:Number, pt:Point = null ):Point
{
if (!pt)
pt = new Point();

pt.x = int(_sceneModel.viewArea.x + (dx / _sceneModel.zoom));
pt.y = int(_sceneModel.viewArea.y + (dy / _sceneModel.zoom));

return pt;
}

protected function onInteraction_mouseDown( x:Number, y:Number, isRightMouse:Boolean =
false ):void
{
_begunInteraction = true;
_entityToFollow = null;
_startDrag.x = x;
_startDrag.y = y;
_residualScroll = false;
_targetViewArea = false;

if (isRightMouse)
_isRightMouseDown = true;
}

protected function hasDragStarted( x:Number, y:Number ):Boolean
{
var isDrag:Boolean;

if (!_stopScroll && (_begunInteraction && (Math.abs(x - _startDrag.x) > THRESHOLD ||
Math.abs(y
```

```
- _startDrag.y) > THRESHOLD)))
isDrag = true;

return isDrag;
}

protected function onInteraction_mouseMove( x:Number, y:Number ):void
{
if (!_dragging)
{
//only start dragging after the player has reached a drag threshold
// if (!_stopScroll && (_begunInteraction && (Math.abs(x - _startDrag.x) > THRESHOLD ||
Math.abs(y - _startDrag.y) > THRESHOLD)))
// _dragging = true;
if (hasDragStarted(x, y))
_dragging = true;

else
return;
}

_scrollDeltaStart.x = _startDrag.x;
_scrollDeltaStart.y = _startDrag.y;
var dx:Number = (_startDrag.x - x) / _sceneModel.zoom;
var dy:Number = (_startDrag.y - y) / _sceneModel.zoom;
adjustLocation(dx, dy);
_startDrag.x = x;
_startDrag.y = y;
}

protected function onInteraction_mouseUp( x:Number, y:Number, isRightMouse:Boolean = false
):void
{
if (_begunInteraction)
{
if (!_dragging)
onClick(x, y);

else
{
_scrollDelta.x = _scrollDeltaStart.x - _startDrag.x;
_scrollDelta.y = _scrollDeltaStart.y - _startDrag.y;

if (Math.abs(_scrollDelta.x) > 10 || Math.abs(_scrollDelta.y) > 10)
_residualScroll = true;
}
}

_begunInteraction = _dragging = false;

if
```

```
(isRightMouse)
_isRightMouseDown = false;
}

public function onInteraction( type:String, x:Number, y:Number ):void
{
var isRightMouseBtn:Boolean = type == MouseEvent.RIGHT_MOUSE_DOWN || type ==
MouseEvent.RIGHT_MOUSE_UP;

switch (type)
{
case MouseEvent.RIGHT_MOUSE_DOWN:
case MouseEvent.MOUSE_DOWN:
{
onInteraction_mouseDown(x, y, isRightMouseBtn);
break;
}

case MouseEvent.MOUSE_MOVE:
{
onInteraction_mouseMove(x, y);
break;
}

case MouseEvent.RIGHT_MOUSE_UP:
case MouseEvent.MOUSE_UP:
{
onInteraction_mouseUp(x, y, isRightMouseBtn);
break;
}
}
}

public function onKey( keyCode:uint, up:Boolean = true ):void
{
switch (keyCode)
{
case KeyboardKey.SUBTRACT.keyCode:
case KeyboardKey.MINUS.keyCode:
onZoom(-.1, Application.STAGE.mouseX, Application.STAGE.mouseY);
break;
case KeyboardKey.ADD.keyCode:
case KeyboardKey.PLUS.keyCode:
onZoom(.1, Application.STAGE.mouseX, Application.STAGE.mouseY);
break;
}
}

protected var _viewController:ViewController;

[Inject]
```

```
public function set viewController( value:ViewController ):void { _viewController = value; }
public function get viewController():ViewController { return _viewController; }

public function onZoom( delta:Number, x:Number, y:Number ):void
{
if (_fteController.running || _viewController.modalHasFocus)
return;

var oldZoom:Number = _sceneModel.zoom;
var newZoom:Number = _sceneModel.zoom + delta;

newZoom = Math.round(newZoom * 10) / 10;
newZoom = newZoom > _maxZoom ? _maxZoom : newZoom;
newZoom = newZoom < _minZoom ? _minZoom : newZoom;

if (newZoom == oldZoom)
return;

var oldDistanceX:Number = _sceneModel.viewArea.x + x / oldZoom;
var oldDistanceY:Number = _sceneModel.viewArea.y + y / oldZoom;
_sceneModel.zoom = newZoom;
var newDistanceX:Number = _sceneModel.viewArea.x + x / newZoom;
var newDistanceY:Number = _sceneModel.viewArea.y + y / newZoom;

adjustLocation(oldDistanceX - newDistanceX, oldDistanceY - newDistanceY);
_interactSignal.zoom(_sceneModel.zoom);
}

protected function onClick( dx:Number, dy:Number ):Vector.<Entity>
{
//see what the player is clicking on
var interacts:Vector.<Entity> = new Vector.<Entity>;
if (_sceneModel.ready)
{
var zoomPoint:Point = new Point(dx / _sceneModel.zoom, dy / _sceneModel.zoom);
var entities:Array = _gridSystem.getEntitiesAt(_sceneModel.viewArea.x + zoomPoint.x,
_sceneModel.viewArea.y + zoomPoint.y);
if (!entities)
return interacts;
var entity:Entity;
var animation:Animation;
_point.setTo(_sceneModel.viewArea.x + zoomPoint.x, _sceneModel.viewArea.y + zoomPoint.y);
_starbaseModel.grid.convertIsoToGrid(_point);
for (var i:int = 0; i < entities.length; i++)
{
entity = Entity(entities[i].entity);
if (entity.has(Interactable))
{
animation = entity.get(Animation);
if
```

```
(!animation.visible)
continue;
hitTest(interacts, entity, animation, zoomPoint.x, zoomPoint.y);
}
}
}
return interacts;
}

protected function hitTest( interacts:Vector.<Entity>, entity:Entity, animation:Animation,
dx:Number, dy:Number ):void
{
var detail:Detail = entity.get(Detail);
var assetVO:AssetVO = detail.assetVO;
switch (detail.category)
{
case CategoryEnum.BUILDING:
case CategoryEnum.STARBASE:
//check if point is within grid bounds of the building
var building:BuildingVO = (entity.has(Building)) ? Building(entity.get(Building)).buildingVO :
Platform(entity.get(Platform)).buildingVO;
_bbox.setTo(building.baseX, building.baseY, building.sizeX, building.sizeY);
if (_bbox.containsPoint(_point))
interacts.push(entity);
break;
default:
_tempPoint.setTo(animation.render.x + animation.offsetX, animation.render.y +
animation.offsetY);
if (assetVO.bbox)
{
//rectangle / point collision
_bbox.setTo(_tempPoint.x + assetVO.bbox.x, _tempPoint.y + assetVO.bbox.y,
assetVO.bbox.width, assetVO.bbox.height);
if (_bbox.contains(dx, dy))
interacts.push(entity);
} else
{
//radius / point collision
_tempPoint.x -= dx;
_tempPoint.y -= dy;
if (((_tempPoint.x * _tempPoint.x) + (_tempPoint.y * _tempPoint.y)) <= (assetVO.radius *
assetVO.radius))
interacts.push(entity);
}
break;
}
}

protected function dispatch( e:Event ):void { _eventDispatcher.dispatchEvent(e); }

protected
```

```
function progressFTE():void
{
inFTE = false;
_fteController.nextStep();
}

/** Doesn't actually tell you if you are in the FTE; this is a flag set by some steps in the FTE to tell us to do something */
public function get inFTE():Boolean { return _inFTE; }
public function set inFTE( v:Boolean ):void { _inFTE = v; _stopScroll = v; }

[Inject]
public function set eventDispatcher( v:IEventDispatcher ):void { _eventDispatcher = v; }
public function set followEntity( entity:Entity ):void { _entityToFollow = entity; }
[Inject]
public function set fteController( value:FTEController ):void { _fteController = value; }
[Inject]
public function set interactSignal( v:InteractSignal ):void { _interactSignal = v; }
[Inject]
public function set keyboardController( v:KeyboardController ):void { _keyboardController = v; }
[Inject]
public function set sceneModel( value:SceneModel ):void { _sceneModel = value; }
public function get sceneX():Number { return (_sceneModel.ready) ? _sceneModel.focus.x : 0; }
public function get sceneY():Number { return (_sceneModel.ready) ? _sceneModel.focus.y : 0; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set viewStack( v:IViewStack ):void { _viewStack = v; }

override public function removeFromGame( game:Game ):void
{
_bbox = null;
if (_controlScheme)
_controlScheme.destroy();
_controlScheme = null;
_entityToFollow = null;

_eventDispatcher.removeEventListener(ViewEvent.SHOW_VIEW, onShowView);
_eventDispatcher = null;

_gridSystem = null;
_interactSignal = null;
_keyboardController = null;
_layer = null;
_point = null;
_presenter = null;
_sceneModel = null;
_startDrag = null;
_targetViewAreaStart = null;
_targetViewAreaDelta = null;
_tempPoint
```

```
= null;
_viewStack = null;
_viewController = null;
}

private function onShowView( event:ViewEvent ):void
{
if (_dragging)
onInteraction_mouseUp(_startDrag.x, _startDrag.y);
}
}
}


-----------------------------------
File 260: igw\com\game\entity\systems\interact\SectorInteractSystem.as
package com.game.entity.systems.interact
{
import com.Application;
import com.controller.ChatController;
import com.controller.GameController;
import com.controller.keyboard.KeyboardKey;
import com.enum.CategoryEnum;
import com.enum.FleetStateEnum;
import com.enum.TypeEnum;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IInteractFactory;
import com.game.entity.nodes.sector.MissionNode;
import com.game.entity.nodes.shared.EnemyNode;
import com.game.entity.nodes.shared.OwnedNode;
import com.game.entity.systems.interact.controls.BrowserScheme;
import com.model.fleet.FleetVO;
import com.model.sector.SectorModel;
import com.presenter.sector.ISectorPresenter;
import com.util.RouteLineBuilder;

import flash.events.MouseEvent;
import flash.geom.Point;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.osflash.signals.Signal;
import org.parade.core.IViewStack;
import
```

```
org.parade.core.ViewController;
import org.parade.enum.ViewEnum;

public class SectorInteractSystem extends InteractSystem
{
[Inject(nodeType="com.game.entity.nodes.sector.MissionNode")]
public var mission:NodeList;
[Inject(nodeType="com.game.entity.nodes.shared.OwnedNode")]
public var owned:NodeList;

public var onCoordsUpdate:Signal;
public var onSelectionChangeSignal:Signal;

private var _chatController:ChatController;
private var _destination:Entity;
private var _game:Game;
private var _gameController:GameController;
private var _interactFactory:IInteractFactory;
private var _missionEntities:Vector.<Entity>;
private var _routeLine:Entity;
private var _sectorModel:SectorModel;
private var _selected:Entity;
private var _selector:Entity;
private var _selectedEnemy:Entity;
private var _selectorEnemy:Entity;

override public function addToGame( game:Game ):void
{
super.addToGame(game);
owned.nodeRemoved.add(onNodeRemoved);
owned.nodeAdded.add(onNodeAdded);
onCoordsUpdate = new Signal(Number, Number);
onSelectionChangeSignal = new Signal(Entity);

_controlScheme = new BrowserScheme();
_controlScheme.init(this, _layer, _keyboardController);
_controlScheme.notifyOnMove = true;

_game = game;
_missionEntities = new Vector.<Entity>;
}

override public function update( time:Number ):void
{
super.update(time);
if (_selected)
{
if (_selectedEnemy && !_selectedEnemy.id)
removeTargetSelection();
if (Move(_selected.get(Move)).moving)
{
```

```
if (_routeLine)
RouteLineBuilder.updateRouteLine(_routeLine, _selected);
if (_entityToFollow)
{
var position:Position = _selected.get(Position);
onCoordsUpdate.dispatch(position.x, position.y);
}
} else
{
if (_destination)
_destination = _interactFactory.destroyInteractEntity(_destination);
if (_routeLine)
_routeLine = _interactFactory.destroyInteractEntity(_routeLine);
}
}
}

override protected function onClick( dx:Number, dy:Number ):Vector.<Entity>
{
//see what the player is clicking on
var attack:Attack;
var interacts:Vector.<Entity> = super.onClick(dx, dy);
var move:Move;
var position:Position;
var entity:Entity;
if (interacts.length > 0)
{
interacts.sort(orderEntities);
for (var i:int = 0; i < interacts.length; i++)
{
entity = interacts[i];
if (entity.has(Owned) && entity.has(Move))
{
if (entity != _selected)
{
selectEntity(entity, false);
onSelectionChangeSignal.dispatch(_selected);
} else
{
presenter.onInteractionWithSectorEntity(dx, dy, entity, _selected);
if (_inFTE)
progressFTE();
}
break;
} else
{
attack = entity.get(Attack);
move = entity.get(Move);
if (move && !_inFTE)
{
```

```
//if (!move.moving)
presenter.onInteractionWithSectorEntity(dx, dy, entity, _selected);
} else
{
if (_inFTE)
{
if (entity.has(Owned))
{
presenter.onInteractionWithSectorEntity(dx, dy, entity, _selected);
progressFTE();
}
} else
presenter.onInteractionWithSectorEntity(dx, dy, entity, _selected);
}
break;
}
}
} else if (_selected)
{
// don't allow the player to command his fleet if it's recalling
var selectedFleetVO:FleetVO = presenter.getFleetVO(_selected.id);
if (selectedFleetVO && !selectedFleetVO.inBattle && selectedFleetVO.state !=
FleetStateEnum.FORCED_RECALLING)
{
//player is trying to move their ship
move = _selected.get(Move);
if (move)
{
position = Position(_selected.get(Position));
var dest:Point = new Point(int(_sceneModel.viewArea.x + (dx / _sceneModel.zoom)),
int(_sceneModel.viewArea.y + (dy / _sceneModel.zoom)));
move.setDestination(dest.x, dest.y);
_gameController.sectorMoveFleet(_selected.id, dest.x, dest.y);
showDestination();
showRouteLine();
}
}
}
return interacts;
}

override public function onInteraction( type:String, x:Number, y:Number ):void
{
super.onInteraction(type, x, y);
if (type == MouseEvent.MOUSE_MOVE)
onShowPosition(null, x, y);
}

override public function onKey( keyCode:uint, up:Boolean = true ):void
{
```

```
if (_chatController.chatHasFocus || _viewController.modalHasFocus)
return;

super.onKey(keyCode, up);

switch (keyCode)
{
case KeyboardKey.A.keyCode:
case KeyboardKey.D.keyCode:
case KeyboardKey.LEFT.keyCode:
case KeyboardKey.RIGHT.keyCode:
{
var cnt:int = 0;
var direction:int = (keyCode == KeyboardKey.A.keyCode || keyCode ==
KeyboardKey.LEFT.keyCode) ? 0 : 1;
var node:OwnedNode;

if (_selected)
{
for (node = owned.head; node; node = node.next)
{
//find the currently selected node
if (node.entity == _selected)
break;
}
}

else if (owned.head)
node = owned.head;

var temp:Entity = (_selected) ? _selected : (node) ? node.entity : null;

while (node && cnt != 1)
{
if (direction == 0)
node = (node.previous) ? node.previous : owned.tail;

else
node = (node.next) ? node.next : owned.head;

if (node.entity == temp)
cnt++;

else if (node.detail.category == CategoryEnum.SHIP)
{
selectEntity(node.entity);
onSelectionChangeSignal.dispatch(_selected);
break;
}
}
```

```
break;
}
}
}

public function showSelector():void
{
if (!_selected)
return;
_selector = _interactFactory.showSelection(_selected, _selector);
if (presenter)
presenter.onBattle();
showDestination();
showRouteLine();
showEnemySelector();
}

private function showDestination():void
{
if (!_selected)
return;
var move:Move = _selected.get(Move);
if (!move)
{
if (_destination)
{
_destination = _interactFactory.destroyInteractEntity(_destination);
}
return;
}

_destination = _interactFactory.showSelection(null, _destination, move.destination.x,
move.destination.y);
}

private function showRouteLine():void
{
if (!_selected)
return;

var move:Move = Move(_selected.get(Move));

if (!move)
{
if (_routeLine)
{
_routeLine = _interactFactory.destroyInteractEntity(_routeLine);
}
return;
```

```
}

// The anim label changing indicates we have selected a new entity
var label:String = _routeLine ? Animation(_routeLine.get(Animation)).label : "";
if (label != (TypeEnum.ROUTE_LINE + _selected.id))
{
if (_routeLine)
{
_routeLine = _interactFactory.destroyInteractEntity(_routeLine);
}
}

if (_routeLine)
{
RouteLineBuilder.adjustRotation(_routeLine, move.destination);
} else
{
_routeLine = _interactFactory.createRouteLine(_selected, move.destination);
}
}

private function showEnemySelector():void
{
if (!_selected)
return;
var attack:Attack = _selected.get(Attack);
if (!attack || attack.targetID == null)
{
_interactFactory.destroyInteractEntity(_selectorEnemy);
_selectorEnemy = _selectedEnemy = null;
return;
}
_selectedEnemy = _game.getEntity(attack.targetID);
if (!_selectedEnemy)
{
_interactFactory.destroyInteractEntity(_selectorEnemy);
_selectorEnemy = _selectedEnemy = null;
return;
}
_selectorEnemy = _interactFactory.showSelection(_selectedEnemy, _selectorEnemy);
_destination = _interactFactory.destroyInteractEntity(_destination);
}

private function onShowPosition( e:MouseEvent = null, x:int = 0, y:int = 0 ):void
{
if (e)
{
x = e.stageX;
y = e.stageY;
}
```

```
if (_sceneModel.viewArea && !_entityToFollow)
onCoordsUpdate.dispatch(_sceneModel.viewArea.x + (x / _sceneModel.zoom),
_sceneModel.viewArea.y + (y / _sceneModel.zoom));
}

private function onNodeRemoved( node:OwnedNode ):void
{
if (node.entity == _selected)
{
_interactFactory.destroyInteractEntity(_selectorEnemy);
_interactFactory.destroyInteractEntity(_destination);
_routeLine = _interactFactory.destroyInteractEntity(_routeLine);
_interactFactory.destroyInteractEntity(_selector);
_selected = _selectedEnemy = _selector = _selectorEnemy = null;
_destination = null;
}
}

private function onNodeAdded( node:OwnedNode ):void
{
if (_selected == null && _sectorModel.focusFleetID != '')
{
if (node.entity.id == _sectorModel.focusFleetID)
selectEntity(node.entity, false);
}
}

private function onEnemyRemoved( node:EnemyNode ):void
{
if (node.entity == _selectedEnemy)
{
_interactFactory.destroyInteractEntity(_selectorEnemy);
_selectedEnemy = _selectorEnemy = null;
}
}

private function orderEntities( entityA:Entity, entityB:Entity ):int
{
var detailA:Detail = entityA.get(Detail);
var detailB:Detail = entityB.get(Detail);
var enemyA:Boolean = entityA.has(Enemy);
var enemyB:Boolean = entityB.has(Enemy);
var ownedA:Boolean = entityA.has(Owned);
var ownedB:Boolean = entityB.has(Owned);

if (detailA.category == CategoryEnum.SECTOR)
return 1;
if (detailB.category == CategoryEnum.SECTOR)
return -1;
return
```

```
0;
}

public function selectEntity( entityToSelect:Entity, gotoLocation:Boolean = true ):void
{
_selected = entityToSelect;
if (_selected)
{
_sectorModel.focusFleetID = _selected.id;
if (gotoLocation)
{
var position:Position = _selected.get(Position);
jumpToLocation(position.x, position.y);
_entityToFollow = (Move(_selected.get(Move)).moving) ? _selected : null;
}
showSelector();
}
}

public function removeTargetSelection():void
{
_selectedEnemy = null;
if (_selected)
{
var attack:Attack = _selected.get(Attack);
if (attack)
attack.targetID = null;
}
showSelector();
}

public function get missionEntities():Vector.<Entity>
{
_missionEntities.length = 0;
var node:MissionNode;
for (node = mission.head; node; node = node.next)
{
_missionEntities.push(node.entity);
}
return _missionEntities;
}

[Inject]
public function set chatController( value:ChatController ):void { _chatController = value; }
[Inject]
public function set gameController( value:GameController ):void { _gameController = value; }
[Inject]
public function set interactFactory( value:IInteractFactory ):void { _interactFactory = value; }
public function set presenter( v:ISectorPresenter ):void { _presenter = v; }
public function get presenter():ISectorPresenter { return ISectorPresenter(_presenter); }
[Inject]
```

```
public function set sectorModel( value:SectorModel ):void { _sectorModel = value; }
public function get selected():Entity { return _selected; }
public function get selectedEnemy():Entity { return _selectedEnemy; }

override public function removeFromGame( game:Game ):void
{
owned.nodeRemoved.remove(onNodeRemoved);
owned.nodeAdded.remove(onNodeAdded);
owned = null;
_interactFactory.destroyInteractEntity(_selectorEnemy);
_interactFactory.destroyInteractEntity(_destination);
_interactFactory.destroyInteractEntity(_selector);
_destination = _selected = _selector = _selectedEnemy = _selectorEnemy = null;
onCoordsUpdate.removeAll();
onCoordsUpdate = null;
onSelectionChangeSignal.removeAll();
onSelectionChangeSignal = null;
super.removeFromGame(game);
_chatController = null;
_game = null;
_gameController = null;
_interactFactory = null;
_missionEntities = null;
_sectorModel = null;
_selected = null;
}
}
}


------------------------------------
File 261: igw\com\game\entity\systems\interact\StarbaseInteractSystem.as
package com.game.entity.systems.interact
{
import com.Application;
import com.controller.ChatController;
import com.controller.keyboard.KeyboardKey;
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.controller.transaction.requirements.UnderMaxCountRequirement;
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.StarbaseConstructionEnum;
import com.enum.TypeEnum;
import com.enum.server.PurchaseTypeEnum;
import com.event.TransactionEvent;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import
```

```
com.game.entity.components.shared.VCList;
import com.game.entity.components.starbase.Platform;
import com.game.entity.factory.IInteractFactory;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.nodes.starbase.BuildingNode;
import com.game.entity.systems.interact.controls.BrowserScheme;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseGrid;
import com.presenter.shared.IUIPresenter;
import com.presenter.starbase.IStarbasePresenter;

import flash.events.MouseEvent;
import flash.geom.Point;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;

public class StarbaseInteractSystem extends InteractSystem
{
public static const BASE_STATE:int = 0;
public static const BUILD_STATE:int = 2;
public static const MOVE_STATE:int = 1;
private static const MAX_FAILED_BUILD_ATTEMPTS:int = 2;

[Inject(nodeType="com.game.entity.nodes.starbase.BuildingNode")]
public var nodes:NodeList;

private static var _buildingID:int = 1;

private var _chatController:ChatController;
private var _crntFailedBuildAttempts:int;
private var _game:Game;
private var _interactFactory:IInteractFactory;
private var _isShiftKeyPressed:Boolean;
private var _purchaseType:uint;
private var _selected:Entity;
private var _starbaseFactory:IStarbaseFactory;
private var _starbaseSystem:StarbaseSystem;
private var _state:int;
private var _tempBaseX:int;
private var _tempBaseY:int;
private var _tempBuildingVO:BuildingVO;
private var _transactionController:TransactionController;
private var _uiPresenter:IUIPresenter;

override
```

```
public function addToGame( game:Game ):void
{
super.addToGame(game);

_controlScheme = new BrowserScheme();
_controlScheme.init(this, _layer, _keyboardController);
_minZoom = 0.383

_game = game;
_starbaseSystem = StarbaseSystem(_game.getSystem(StarbaseSystem));
nodes.nodeRemoved.add(onNodeRemoved);

//left bounding line
_starbaseFactory.createBoundingLine(StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT,
StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH, StarbaseGrid.PLAY_SPACE_HEIGHT -
StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT);
//top bounding line
_starbaseFactory.createBoundingLine(StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT,
StarbaseGrid.PLAY_SPACE_WIDTH - StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT);
//right bound line
_starbaseFactory.createBoundingLine(StarbaseGrid.PLAY_SPACE_WIDTH -
StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT,
StarbaseGrid.PLAY_SPACE_WIDTH - StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_HEIGHT - StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT);
//bottom bounding line
_starbaseFactory.createBoundingLine(StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_HEIGHT - StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT,
StarbaseGrid.PLAY_SPACE_WIDTH - StarbaseGrid.PLAY_SPACE_NOBUILD_WIDTH,
StarbaseGrid.PLAY_SPACE_HEIGHT - StarbaseGrid.PLAY_SPACE_NOBUILD_HEIGHT);
}

override public function onInteraction( type:String, x:Number, y:Number ):void
{
super.onInteraction(type, x, y);

if (!_selected || !_sceneModel.ready || type != MouseEvent.MOUSE_MOVE)
return;

onMoveBuildingComponent(null, x, y);
}

override public function onKey( keyCode:uint, up:Boolean = true ):void
{
if (_chatController.chatHasFocus || _viewController.modalHasFocus)
return;

super.onKey(keyCode,
```

```
up);

if (keyCode == KeyboardKey.ESCAPE.keyCode && (_state == BUILD_STATE || _state ==
MOVE_STATE) && !_inFTE)
resolveBuilding(false);

if (keyCode == KeyboardKey.SHIFT.keyCode)
_isShiftKeyPressed = !up;
}

override protected function onClick( dx:Number, dy:Number ):Vector.<Entity>
{
//see what the player is clicking on
var interacts:Vector.<Entity> = super.onClick(dx, dy);
var len:uint = interacts.length;
var detail:Detail;
var wasPlaced:Boolean;

switch (_state)
{
case BASE_STATE:
_selected = null;
if (len > 0)
{
var currentEntity:Entity;
for (var i:uint = 0; i < len; ++i)
{
currentEntity = interacts[i];
detail = currentEntity.get(Detail);
//determine if the player is clicking a building or a starbase platform
if (detail.prototypeVO.getValue('constructionCategory') ==
StarbaseConstructionEnum.PLATFORM)
{
//only move the platform if there is not a building sitting atop of it
if (_starbaseModel.grid.canMovePlatform(Platform(currentEntity.get(Platform)).buildingVO))
_selected = currentEntity;
} else
{
//Set the new selected
_selected = currentEntity;
break;
}
}

if (_selected && _selected.id.indexOf('clientside_') == -1)
{
presenter.onInteractionWithBaseEntity(dx, dy, _selected);
if (_inFTE)
progressFTE();
updateRanges();
}
```

```
}
break;
case MOVE_STATE:
case BUILD_STATE:
if (_selected)
{
var failed:Boolean;
wasPlaced = placeBuilding(_selected);
if (!wasPlaced)
{
if (_crntFailedBuildAttempts < MAX_FAILED_BUILD_ATTEMPTS)
{
//do we want to do this during FTE?
if (!_inFTE)
_crntFailedBuildAttempts++;
} else //abort build logic for too many failed attempts
failed = true;
}

if (wasPlaced || failed)
{
if (wasPlaced && _inFTE)
progressFTE();
resolveBuilding(wasPlaced);
}

if (wasPlaced && _isShiftKeyPressed)
{
//determine if the previous build requesst is a platform, since this is the only type we're allowing
shift-click building for mutil units
var isPlatform:Boolean;

if (!_tempBuildingVO || !_tempBuildingVO.prototype)
isPlatform = false;
else
isPlatform = _tempBuildingVO.constructionCategory ==
StarbaseConstructionEnum.PLATFORM;

if (isPlatform)
{
var requirements:RequirementVO =
_transactionController.canBuild(_tempBuildingVO.prototype);
if (requirements.purchaseVO.alloyAmountShort == 0 &&
requirements.purchaseVO.creditsAmountShort == 0 &&
requirements.purchaseVO.energyAmountShort == 0 &&
requirements.purchaseVO.syntheticAmountShort == 0 &&
requirements.getRequirementOfType(UnderMaxCountRequirement).isMet)
{
presenter.performTransaction(TransactionEvent.STARBASE_BUILDING_BUILD,
_tempBuildingVO.prototype,
```

```
_purchaseType);
onMoveBuildingComponent(null, Application.STAGE.mouseX, Application.STAGE.mouseY);
}
}
}
}

break;
}
if (_selected == null)
updateRanges();
return interacts;
}

private function onNodeRemoved( node:BuildingNode ):void
{
if (_selected == node.entity)
{
_selected = null;
updateRanges();
}
}

private function resolveBuilding( placed:Boolean ):void
{
if (!_selected)
return;

if (_selected.get(Platform))
Animation(_selected.get(Animation)).render.color = 0xffffff;

if (placed)
_gridSystem.forceGridCheck(_selected);
else
{
var position:Position = _selected.get(Position);
var vo:BuildingVO = _starbaseModel.getBuildingByID(_selected.id);

if (_state == BUILD_STATE)
_starbaseFactory.destroyStarbaseItem(_selected);
else
{
//player was trying to move the building
//put the building back where it was before we started the move process
vo = _starbaseModel.getBuildingByID(_selected.id);
vo.baseX = _tempBaseX;
vo.baseY = _tempBaseY;
position = _selected.get(Position);
_starbaseModel.grid.convertBuildingGridToIso(position.position, vo);
//have to do this to update followers
position.x
```

```
= position.position.x;
position.y = position.position.y;
_starbaseModel.grid.addToGrid(vo);
_starbaseSystem.depthSort(StarbaseSystem.DEPTH_SORT_ALL);
_gridSystem.forceGridCheck(_selected);

if (vo.itemClass == TypeEnum.PYLON)
{
_starbaseSystem.positionPylonBase(_selected);
_starbaseSystem.findPylonConnections(_selected);
}
}
}

//reset everythings
_crntFailedBuildAttempts = 0;
_selected = null;
_starbaseSystem.showShields(true);
setState(BASE_STATE);
updateRanges();
}


//==========================================================================================
//******************************************************************************************
// Building and Moving
//******************************************************************************************

//==========================================================================================

private function placeBuilding( entity:Entity ):Boolean
{
/*var selectedPosition:Position = entity.get(Position);
var p:Point = new Point(selectedPosition.x, selectedPosition.y);
_starbaseModel.grid.convertIsoToGrid(p);
trace(p.toString());
trace(_tempBuildingVO.baseX, _tempBuildingVO.baseY);*/

var placed:Boolean = false;
var result:int = _starbaseModel.grid.confirmAddToGrid(_tempBuildingVO);
if (result == StarbaseGrid.ALL_CLEAR)
{
//update the buildingVO with the new position
var buildingVO:BuildingVO = _tempBuildingVO;
if (entity.id.indexOf('clientside_') == -1)
{
_starbaseModel.grid.addToGrid(buildingVO);
_transactionController.starbaseMoveBuilding(buildingVO, _tempBaseX, _tempBaseY);
} else
{
buildingVO.popuplateFromEntity(entity);
```

```
buildingVO.baseID = _starbaseModel.currentBaseID;

_transactionController.starbaseBuild(buildingVO, _purchaseType);
_starbaseModel.addBuilding(buildingVO);
_purchaseType = PurchaseTypeEnum.NORMAL;
}
_starbaseSystem.findPylonConnections();
placed = true;
}
//update the buildingVO of the entity
if (placed)
{
if (entity.get(Platform))
Animation(entity.get(Animation)).render.color = 0xffffff;
_starbaseSystem.depthSort(StarbaseSystem.DEPTH_SORT_ALL);
}
return placed;
}

private function onMoveBuildingComponent( e:MouseEvent = null, x:int = 0, y:int = 0 ):void
{
if (e)
{
x = e.stageX;
y = e.stageY;
}
if ((_state == MOVE_STATE || _state == BUILD_STATE) && _selected)
{
var result:int = StarbaseGrid.GRID_OCCUPIED;
var animation:Animation;
var position:Position = _selected.get(Position);
var currentDetail:Detail = _selected.get(Detail);

var px:Number = position.x;
var py:Number = position.y;
position.x = _sceneModel.viewArea.x + x / _sceneModel.zoom;
position.y = _sceneModel.viewArea.y + y / _sceneModel.zoom;
_starbaseModel.grid.snapToGrid(position.position, _tempBuildingVO, currentDetail.category ==
CategoryEnum.BUILDING && currentDetail.prototypeVO.itemClass != TypeEnum.PYLON);

//exit out if the grid position of this entity did not change to avoid depth sorting and all that other
expensive stuff
if (position.x == px && position.y == py)
{
//need to do this so other entities attached to this position component are updated
position.x = position.position.x;
position.y = position.position.y;
return;
}

if
```

```
(currentDetail.category == CategoryEnum.STARBASE)
{
animation = _selected.get(Animation);
if (animation.render)
{
result = _starbaseModel.grid.confirmAddToGrid(_tempBuildingVO);
animation.render.color = (result == StarbaseGrid.ALL_CLEAR) ? 0x00ff00 : 0xff0000;
}
} else
{
//see if the selected item can be placed. update the iso square to reflect
var iso:Entity =
VCList(_selected.get(VCList)).getComponent(getIsoSquareType(_tempBuildingVO.prototype));
if (iso)
{
animation = iso.get(Animation);
if (animation.render)
{
result = _starbaseModel.grid.confirmAddToGrid(_tempBuildingVO);
animation.render.color = (result == StarbaseGrid.ALL_CLEAR) ? 0x00ff00 : 0xff0000;
}
}
}
if (result == StarbaseGrid.ALL_CLEAR)
{
_starbaseSystem.depthSort((currentDetail.category == CategoryEnum.STARBASE) ?
StarbaseSystem.DEPTH_SORT_PLATFORMS :
StarbaseSystem.DEPTH_SORT_BUILDINGS);
} else
Position(_selected.get(Position)).depth = 9000;

switch (currentDetail.type)
{
case TypeEnum.POINT_DEFENSE_PLATFORM:
_starbaseSystem.findPylonConnections(null);
break;
case TypeEnum.SHIELD_GENERATOR:
_starbaseSystem.findPylonConnections(null);
_starbaseSystem.showShields(false, _selected);
break;
case TypeEnum.PYLON:
_starbaseSystem.positionPylonBase(_selected);
_starbaseSystem.findPylonConnections(_selected);
break;
default:
_starbaseSystem.findPylonConnections();
_starbaseSystem.showShields();
break;
}

//need
```

```
to do this so other entities attached to this position component are updated
position.x = position.position.x;
position.y = position.position.y;
}
}

public function buildFromPrototype( buildingPrototype:IPrototype, purchaseType:uint ):void
{
// note that the server will reply with the ACTUAL name of the building
_purchaseType = purchaseType;
_tempBuildingVO = new BuildingVO();
_tempBuildingVO.prototype = buildingPrototype;
var id:String = buildingID;
var p:Point = new Point(_sceneModel.focus.x, _sceneModel.focus.y);
switch (_tempBuildingVO.constructionCategory)
{
case StarbaseConstructionEnum.PLATFORM:
_starbaseModel.grid.snapToGrid(p, _tempBuildingVO, false);
_selected = _starbaseFactory.createBaseItem(id, _tempBuildingVO);
break;
default:
_starbaseModel.grid.snapToGrid(p, _tempBuildingVO);
_selected = _starbaseFactory.createBuilding(id, _tempBuildingVO);
break;
}
Position(_selected.get(Position)).depth = -1;
setState(BUILD_STATE);
}

public function setState( newState:int ):void
{
if (newState == MOVE_STATE && _selected == null)
return;
_state = newState;
_controlScheme.notifyOnMove = false;
switch (_state)
{
case BASE_STATE:
_uiPresenter.hudEnabled = true;
break;

case MOVE_STATE:
//if we're moving then remove the building from the grid
if (_selected)
{
_tempBuildingVO = _starbaseModel.getBuildingByID(_selected.id);
_tempBaseX = _tempBuildingVO.baseX;
_tempBaseY = _tempBuildingVO.baseY;
_starbaseModel.grid.removeFromGrid(_tempBuildingVO);
}

//
```

Intentional fallthrough

```
case BUILD_STATE:
if (_selected)
{
_controlScheme.notifyOnMove = true;
Position(_selected.get(Position)).depth = -1;
_uiPresenter.hudEnabled = false;
//Remove any previous ranges and display all ranges during the move
updateRanges();
}
break;
}

//cycle through the existing buildings and add or remove the iso squares from beneath them
var node:BuildingNode;
var selectedDetail:Detail = (_selected) ? _selected.get(Detail) : null;
var type:String;
var vcList:VCList;
for (node = nodes.head; node; node = node.next)
{
vcList = node.entity.get(VCList);
if (vcList)
{
type = getIsoSquareType(node.detail.prototypeVO);
if (_state == BASE_STATE)
vcList.removeComponentType(type);
else if (_selected)
{
if ((selectedDetail.type != TypeEnum.PYLON && Detail(node.entity.get(Detail)).type !=
TypeEnum.PYLON) ||
(selectedDetail.type == TypeEnum.PYLON && Detail(node.entity.get(Detail)).type ==
TypeEnum.PYLON))
vcList.addComponentType(type);
}
}
}
}

public function updateRanges():void
{
if (_selected)
{
var detail:Detail = _selected.get(Detail);
//cycle through the existing buildings and add ranges to those that need
if (_state != MOVE_STATE)
{
_starbaseSystem.showPylonRanges(null, false);
_starbaseSystem.showShieldRanges(null, false);
_starbaseSystem.showTurretRanges(null, false);
_starbaseSystem.showShields(true);
```

```
switch (detail.type)
{
case TypeEnum.POINT_DEFENSE_PLATFORM:
_starbaseSystem.showTurretRanges(_selected, true);
break;
case TypeEnum.PYLON:
_starbaseSystem.showPylonRanges(_selected, true);
break;
case TypeEnum.SHIELD_GENERATOR:
_starbaseSystem.showShieldRanges(_selected, true);
_starbaseSystem.showShields(false, _selected);
break;
}
} else
{
switch (detail.type)
{
case TypeEnum.POINT_DEFENSE_PLATFORM:
_starbaseSystem.showTurretRanges(null, true);
break;
case TypeEnum.PYLON:
_starbaseSystem.showPylonRanges(null, true);
break;
case TypeEnum.SHIELD_GENERATOR:
_starbaseSystem.showShieldRanges(null, true);
_starbaseSystem.showShields();
break;
default:
_starbaseSystem.showShieldRanges(null, true);
_starbaseSystem.showTurretRanges(null, true);
_starbaseSystem.showShields();
break;
}
}
} else
{
_starbaseSystem.showPylonRanges(null, false);
_starbaseSystem.showShieldRanges(null, false);
_starbaseSystem.showTurretRanges(null, false);
_starbaseSystem.showShields(true);
}
}

private function getIsoSquareType( prototypeVO:IPrototype ):String
{
switch (prototypeVO.getValue('sizeX'))
{
case 5:
if (CurrentUser.faction == FactionEnum.IGA)
return
```

```
TypeEnum.ISO_1x1_IGA;
else if (CurrentUser.faction == FactionEnum.SOVEREIGNTY)
return TypeEnum.ISO_1x1_SOVEREIGNTY;
return TypeEnum.ISO_1x1_TYRANNAR;
case 10:
if (CurrentUser.faction == FactionEnum.IGA)
return TypeEnum.ISO_2x2_IGA;
else if (CurrentUser.faction == FactionEnum.SOVEREIGNTY)
return TypeEnum.ISO_2x2_SOVEREIGNTY;
return TypeEnum.ISO_2x2_TYRANNAR;
}
if (CurrentUser.faction == FactionEnum.IGA)
return TypeEnum.ISO_3x3_IGA;
else if (CurrentUser.faction == FactionEnum.SOVEREIGNTY)
return TypeEnum.ISO_3x3_SOVEREIGNTY;
return TypeEnum.ISO_3x3_TYRANNAR;
}

public function get buildingID():String { ++_buildingID; return CurrentUser.name +
'.clientside_building.' + String(_buildingID); }
[Inject]
public function set chatController( v:ChatController ):void { _chatController = v; }
[Inject]
public function set interactFactory( v:IInteractFactory ):void { _interactFactory = v; }
public function set presenter( v:IStarbasePresenter ):void { _presenter = v; }
public function get presenter():IStarbasePresenter { return IStarbasePresenter(_presenter); }
[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set uiPresenter( value:IUIPresenter ):void { _uiPresenter = value; }

override public function removeFromGame( game:Game ):void
{
nodes.nodeRemoved.remove(onNodeRemoved);
super.removeFromGame(game);

_chatController = null;
_selected = null;
_interactFactory.clearRanges();
_interactFactory = null;
_game = null;
_starbaseFactory = null;
_starbaseSystem = null;
_tempBuildingVO = null;
_transactionController = null;
_uiPresenter = null;
}
}
```

```
}


------------------------------------
File 262: igw\com\game\entity\systems\interact\controls\BrowserScheme.as
package com.game.entity.systems.interact.controls
{
import com.Application;
import com.controller.keyboard.KeyboardController;
import com.controller.keyboard.KeyboardKey;
import com.event.StateEvent;
import com.game.entity.systems.interact.InteractSystem;

import flash.display.Stage;
import flash.events.MouseEvent;
import flash.external.ExternalInterface;

import org.starling.events.Touch;
import org.starling.events.TouchEvent;
import org.starling.events.TouchPhase;

public class BrowserScheme implements IControlScheme
{
public static const ALLOW_ALL:int = 0;
public static const LIMITED_ACCESS:int = 1;

private var _interactSystem:InteractSystem;
private var _keyboardController:KeyboardController;
private var _layer:*;
private var _notifyOnMove:Boolean = false;
private var _state:String;

public function init( interactSystem:InteractSystem, layer:*, keyController:KeyboardController
):void
{
_interactSystem = interactSystem;
_keyboardController = keyController;
_layer = layer;
_state = Application.STATE;
addListeners();

addUpKey(KeyboardKey.SUBTRACT.keyCode);
addUpKey(KeyboardKey.MINUS.keyCode);
addUpKey(KeyboardKey.ADD.keyCode);
addUpKey(KeyboardKey.PLUS.keyCode);

if (Application.STATE == StateEvent.GAME_BATTLE_INIT || Application.STATE ==
StateEvent.GAME_BATTLE)
{
addUpKey(KeyboardKey.ONE.keyCode);
```

```
addUpKey(KeyboardKey.TWO.keyCode);
addUpKey(KeyboardKey.THREE.keyCode);
addUpKey(KeyboardKey.FOUR.keyCode);
addUpKey(KeyboardKey.FIVE.keyCode);
addUpKey(KeyboardKey.SIX.keyCode);
addUpKey(KeyboardKey.SEVEN.keyCode);
addUpKey(KeyboardKey.EIGHT.keyCode);
addUpKey(KeyboardKey.NINE.keyCode);
addUpKey(KeyboardKey.ZERO.keyCode);

addUpKey(KeyboardKey.F1.keyCode);
addUpKey(KeyboardKey.F2.keyCode);
addUpKey(KeyboardKey.F3.keyCode);
addUpKey(KeyboardKey.F4.keyCode);
addUpKey(KeyboardKey.F5.keyCode);
addUpKey(KeyboardKey.F6.keyCode);

addUpKey(KeyboardKey.A.keyCode);
addUpKey(KeyboardKey.D.keyCode);
addUpKey(KeyboardKey.DOWN.keyCode);
addUpKey(KeyboardKey.LEFT.keyCode);
addUpKey(KeyboardKey.RIGHT.keyCode);
addUpKey(KeyboardKey.Q.keyCode);
addUpKey(KeyboardKey.S.keyCode);
addUpKey(KeyboardKey.UP.keyCode);
addUpKey(KeyboardKey.W.keyCode);
addUpKey(KeyboardKey.F.keyCode);
addUpKey(KeyboardKey.G.keyCode);
addUpKey(KeyboardKey.T.keyCode);
addUpKey(KeyboardKey.R.keyCode);

addUpKey(KeyboardKey.ESCAPE.keyCode);
addDownKey(KeyboardKey.CONTROL.keyCode);
addUpKey(KeyboardKey.CONTROL.keyCode);
addDownKey(KeyboardKey.SHIFT.keyCode);
addUpKey(KeyboardKey.SHIFT.keyCode);
}

else if (Application.STATE == StateEvent.GAME_SECTOR_INIT)
{
addUpKey(KeyboardKey.A.keyCode);
addUpKey(KeyboardKey.D.keyCode);
addUpKey(KeyboardKey.LEFT.keyCode);
addUpKey(KeyboardKey.RIGHT.keyCode);
}

else
{
addDownKey(KeyboardKey.SHIFT.keyCode);
addUpKey(KeyboardKey.SHIFT.keyCode);
```

```
addUpKey(KeyboardKey.ESCAPE.keyCode);
}

if (ExternalInterface.available)
ExternalInterface.addCallback("onMouseWheel_jsCallback", onMouseWheel_jsCallback);
}

public function addDownKey( keyCode:uint ):void {
_keyboardController.addKeyDownListener(onDownKey, keyCode); }
public function removeDownKey( keyCode:uint ):void {
_keyboardController.removeKeyDownListener(onDownKey, keyCode); }
public function addUpKey( keyCode:uint ):void {
_keyboardController.addKeyUpListener(onUpKey, keyCode); }
public function removeUpKey( keyCode:uint ):void {
_keyboardController.removeKeyUpListener(onUpKey, keyCode); }

protected function onDownKey( keyCode:uint ):void { _interactSystem.onKey(keyCode, false); }
protected function onUpKey( keyCode:uint ):void { _interactSystem.onKey(keyCode, true); }

protected function onTouch( e:TouchEvent ):void
{
var touch:Touch = e.getTouch(_layer);
if (!touch)
return;

switch (touch.phase)
{
case TouchPhase.BEGAN:
_interactSystem.onInteraction(MouseEvent.MOUSE_DOWN, touch.globalX, touch.globalY);
break;
case TouchPhase.HOVER:
if (_notifyOnMove)
_interactSystem.onInteraction(MouseEvent.MOUSE_MOVE, touch.globalX, touch.globalY);
break;
case TouchPhase.MOVED:
_interactSystem.onInteraction(MouseEvent.MOUSE_MOVE, touch.globalX, touch.globalY);
break;
case TouchPhase.ENDED:
_interactSystem.onInteraction(MouseEvent.MOUSE_UP, touch.globalX, touch.globalY);
break;
}
}

protected function onMouse( e:MouseEvent ):void
{
switch (e.type)
{
case MouseEvent.MOUSE_DOWN:
case MouseEvent.RIGHT_MOUSE_DOWN:
Application.STAGE.addEventListener(MouseEvent.MOUSE_MOVE,
```

```
onMouse);
Application.STAGE.addEventListener(MouseEvent.MOUSE_UP, onMouse);
_interactSystem.onInteraction(MouseEvent.MOUSE_DOWN, e.stageX, e.stageY);
break;
case MouseEvent.MOUSE_MOVE:
_interactSystem.onInteraction(MouseEvent.MOUSE_MOVE, e.stageX, e.stageY);
break;
case MouseEvent.MOUSE_UP:
case MouseEvent.RIGHT_MOUSE_UP:
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_UP, onMouse);
_interactSystem.onInteraction(MouseEvent.MOUSE_UP, e.stageX, e.stageY);
break;
}
}

private function onMouseWheel_jsCallback( delta:Number ):void
{
var event:MouseEvent = new MouseEvent(MouseEvent.MOUSE_WHEEL, true, true,
Application.STAGE.mouseX, Application.STAGE.mouseY, Application.STAGE, false, false,
false, false, delta);
Application.STAGE.dispatchEvent(event);
}

protected function onMouseWheel( e:MouseEvent ):void
{
var stage:Stage = Application.STAGE;
_interactSystem.onZoom(e.delta > 0 ? .1 : -.1, e.stageX, e.stageY);
}

protected function addListeners():void
{
if (Application.STARLING_ENABLED)
_layer.addEventListener(TouchEvent.TOUCH, onTouch);

else
{
_layer.addEventListener(MouseEvent.MOUSE_DOWN, onMouse);
_layer.addEventListener(MouseEvent.MOUSE_UP, onMouse);
}

Application.STAGE.addEventListener(MouseEvent.RIGHT_MOUSE_DOWN, onMouse);
Application.STAGE.addEventListener(MouseEvent.RIGHT_MOUSE_UP, onMouse);
Application.STAGE.addEventListener(MouseEvent.MOUSE_WHEEL, onMouseWheel);
}

protected function removeListeners():void
{
if (Application.STARLING_ENABLED)
_layer.removeEventListener(TouchEvent.TOUCH, onTouch);

else
```

```
{
_layer.removeEventListener(MouseEvent.MOUSE_DOWN, onMouse);
_layer.removeEventListener(MouseEvent.MOUSE_UP, onMouse);
}

Application.STAGE.removeEventListener(MouseEvent.RIGHT_MOUSE_DOWN, onMouse);
Application.STAGE.removeEventListener(MouseEvent.RIGHT_MOUSE_UP, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_UP, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_WHEEL, onMouseWheel);
}

public function set notifyOnMove( v:Boolean ):void
{
_notifyOnMove = v;

if (!Application.STARLING_ENABLED)
{
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouse);

if (v)
Application.STAGE.addEventListener(MouseEvent.MOUSE_MOVE, onMouse);
}
}

public function destroy():void
{
removeUpKey(KeyboardKey.SUBTRACT.keyCode);
removeUpKey(KeyboardKey.MINUS.keyCode);
removeUpKey(KeyboardKey.ADD.keyCode);
removeUpKey(KeyboardKey.PLUS.keyCode);
if (_state == StateEvent.GAME_BATTLE_INIT || _state == StateEvent.GAME_BATTLE || _state
== StateEvent.GAME_BATTLE_CLEANUP)
{
removeUpKey(KeyboardKey.ONE.keyCode);
removeUpKey(KeyboardKey.TWO.keyCode);
removeUpKey(KeyboardKey.THREE.keyCode);
removeUpKey(KeyboardKey.FOUR.keyCode);
removeUpKey(KeyboardKey.FIVE.keyCode);
removeUpKey(KeyboardKey.SIX.keyCode);
removeUpKey(KeyboardKey.SEVEN.keyCode);
removeUpKey(KeyboardKey.EIGHT.keyCode);
removeUpKey(KeyboardKey.NINE.keyCode);
removeUpKey(KeyboardKey.ZERO.keyCode);

removeUpKey(KeyboardKey.F1.keyCode);
removeUpKey(KeyboardKey.F2.keyCode);
removeUpKey(KeyboardKey.F3.keyCode);
removeUpKey(KeyboardKey.F4.keyCode);
removeUpKey(KeyboardKey.F5.keyCode);
```

```
removeUpKey(KeyboardKey.F6.keyCode);

removeUpKey(KeyboardKey.A.keyCode);
removeUpKey(KeyboardKey.D.keyCode);
removeUpKey(KeyboardKey.DOWN.keyCode);
removeUpKey(KeyboardKey.LEFT.keyCode);
removeUpKey(KeyboardKey.RIGHT.keyCode);
removeUpKey(KeyboardKey.Q.keyCode);
removeUpKey(KeyboardKey.S.keyCode);
removeUpKey(KeyboardKey.UP.keyCode);
removeUpKey(KeyboardKey.W.keyCode);
removeUpKey(KeyboardKey.F.keyCode);
removeUpKey(KeyboardKey.G.keyCode);
removeUpKey(KeyboardKey.T.keyCode);
removeUpKey(KeyboardKey.R.keyCode);

removeUpKey(KeyboardKey.ESCAPE.keyCode);
removeDownKey(KeyboardKey.CONTROL.keyCode);
removeUpKey(KeyboardKey.CONTROL.keyCode);
removeDownKey(KeyboardKey.SHIFT.keyCode);
removeUpKey(KeyboardKey.SHIFT.keyCode);
} else if (_state == StateEvent.GAME_SECTOR_INIT || _state == StateEvent.GAME_SECTOR
|| _state == StateEvent.GAME_SECTOR_CLEANUP)
{
removeUpKey(KeyboardKey.A.keyCode);
removeUpKey(KeyboardKey.D.keyCode);
removeUpKey(KeyboardKey.LEFT.keyCode);
removeUpKey(KeyboardKey.RIGHT.keyCode);
} else
{
removeDownKey(KeyboardKey.SHIFT.keyCode);
removeUpKey(KeyboardKey.SHIFT.keyCode);
removeUpKey(KeyboardKey.ESCAPE.keyCode);
}

removeListeners();
_interactSystem = null;
_keyboardController = null;
_layer = null;
_notifyOnMove = false;
}
}
}

------------------------------------
File 263: igw\com\game\entity\systems\interact\controls\ControlledEntity.as
package com.game.entity.systems.interact.controls
{
import org.ash.core.Entity;

public
```

```
class ControlledEntity
{
public var entity:Entity;
public var range:Entity;
public var selector:Entity;
public var selectedEnemy:Entity;

private var _destination:SelectorEntity;
private var _selectorEnemy:SelectorEntity;

public function get destination():SelectorEntity { return _destination; }
public function set destination( v:SelectorEntity ):void
{
if (_destination == v)
return;
if (_destination)
_destination.decreaseCount();
_destination = v;
if (_destination != null)
_destination.increaseCount();
}

public function get selectorEnemy():SelectorEntity { return _selectorEnemy; }
public function set selectorEnemy( v:SelectorEntity ):void
{
if (_selectorEnemy == v)
return;
if (_selectorEnemy)
_selectorEnemy.decreaseCount();
_selectorEnemy = v;
if (_selectorEnemy != null)
_selectorEnemy.increaseCount();
}

public function destroy():void
{
destination = null;
entity = null;
range = null;
selector = null;
selectedEnemy = null;
selectorEnemy = null;
}
}
}

-----------------------------------
File 264: igw\com\game\entity\systems\interact\controls\IControlScheme.as
package com.game.entity.systems.interact.controls
{
import
```

```
com.controller.keyboard.KeyboardController;
import com.game.entity.systems.interact.InteractSystem;

public interface IControlScheme
{
function init( interactSystem:InteractSystem, layer:*, keyController:KeyboardController ):void;

function addDownKey( keyCode:uint ):void;
function removeDownKey( keyCode:uint ):void;
function addUpKey( keyCode:uint ):void;
function removeUpKey( keyCode:uint ):void;

function set notifyOnMove( v:Boolean ):void;

function destroy():void;
}
}
```

File 265: igw\com\game\entity\systems\interact\controls\MobileScheme.as

```
package com.game.entity.systems.interact.controls
{
import com.Application;
import com.controller.keyboard.KeyboardController;
import com.controller.keyboard.KeyboardKey;
import com.event.StateEvent;
import com.game.entity.systems.interact.InteractSystem;

import flash.events.MouseEvent;
import flash.geom.Point;

import org.parade.util.DeviceMetrics;
import org.starling.events.Touch;
import org.starling.events.TouchEvent;
import org.starling.events.TouchPhase;

public class MobileScheme implements IControlScheme
{
public static const ALLOW_ALL:int = 0;
public static const LIMITED_ACCESS:int = 1;

private var _distance:Number = 0;
private var _interactSystem:InteractSystem;
private var _keyboardController:KeyboardController;
private var _layer:*;
private var _notifyOnMove:Boolean = false;
private var _p1:Point;
private var _p2:Point;
private var _state:String;
private var _touches:Vector.<Touch>;
private
```

```
var _zooming:Boolean = false;

public function init( interactSystem:InteractSystem, layer:*, keyController:KeyboardController
):void
{
_interactSystem = interactSystem;
_keyboardController = keyController;
_layer = layer;
_p1 = new Point();
_p2 = new Point();
_state = Application.STATE;
_touches = new Vector.<Touch>;
addListeners();

if (Application.STATE == StateEvent.GAME_BATTLE_INIT || Application.STATE ==
StateEvent.GAME_BATTLE)
{
addUpKey(KeyboardKey.ONE.keyCode);
addUpKey(KeyboardKey.TWO.keyCode);
addUpKey(KeyboardKey.THREE.keyCode);
addUpKey(KeyboardKey.FOUR.keyCode);
addUpKey(KeyboardKey.FIVE.keyCode);
addUpKey(KeyboardKey.SIX.keyCode);
addUpKey(KeyboardKey.SEVEN.keyCode);
addUpKey(KeyboardKey.EIGHT.keyCode);
addUpKey(KeyboardKey.NINE.keyCode);
addUpKey(KeyboardKey.ZERO.keyCode);

addUpKey(KeyboardKey.F1.keyCode);
addUpKey(KeyboardKey.F2.keyCode);
addUpKey(KeyboardKey.F3.keyCode);
addUpKey(KeyboardKey.F4.keyCode);
addUpKey(KeyboardKey.F5.keyCode);
addUpKey(KeyboardKey.F6.keyCode);

addUpKey(KeyboardKey.A.keyCode);
addUpKey(KeyboardKey.D.keyCode);
addUpKey(KeyboardKey.DOWN.keyCode);
addUpKey(KeyboardKey.LEFT.keyCode);
addUpKey(KeyboardKey.RIGHT.keyCode);
addUpKey(KeyboardKey.Q.keyCode);
addUpKey(KeyboardKey.S.keyCode);
addUpKey(KeyboardKey.UP.keyCode);
addUpKey(KeyboardKey.W.keyCode);
addUpKey(KeyboardKey.F.keyCode);
addUpKey(KeyboardKey.G.keyCode);
addUpKey(KeyboardKey.T.keyCode);
addUpKey(KeyboardKey.R.keyCode);

addUpKey(KeyboardKey.ESCAPE.keyCode);
addDownKey(KeyboardKey.CONTROL.keyCode);
```

```
addUpKey(KeyboardKey.CONTROL.keyCode);
addDownKey(KeyboardKey.SHIFT.keyCode);
addUpKey(KeyboardKey.SHIFT.keyCode);
} else if (Application.STATE == StateEvent.GAME_SECTOR_INIT)
{
addUpKey(KeyboardKey.A.keyCode);
addUpKey(KeyboardKey.D.keyCode);
addUpKey(KeyboardKey.LEFT.keyCode);
addUpKey(KeyboardKey.RIGHT.keyCode);
} else
{
addDownKey(KeyboardKey.SHIFT.keyCode);
addUpKey(KeyboardKey.SHIFT.keyCode);
addUpKey(KeyboardKey.ESCAPE.keyCode);
}
}

public function addDownKey( keyCode:uint ):void {
_keyboardController.addKeyDownListener(onDownKey, keyCode); }
public function removeDownKey( keyCode:uint ):void {
_keyboardController.removeKeyDownListener(onDownKey, keyCode); }
public function addUpKey( keyCode:uint ):void {
_keyboardController.addKeyUpListener(onUpKey, keyCode); }
public function removeUpKey( keyCode:uint ):void {
_keyboardController.removeKeyUpListener(onUpKey, keyCode); }

protected function onDownKey( keyCode:uint ):void { _interactSystem.onKey(keyCode, false); }
protected function onUpKey( keyCode:uint ):void { _interactSystem.onKey(keyCode, true); }

protected function onTouch( e:TouchEvent ):void
{
var touch:Touch = e.getTouch(_layer);
if (!touch)
return;
_touches.length = 0;
e.getTouches(_layer, null, _touches);
switch (touch.phase)
{
case TouchPhase.BEGAN:
if (_touches.length == 2)
{
_zooming = false;
_p1.setTo(_touches[0].globalX, _touches[0].globalY);
_p2.setTo(_touches[1].globalX, _touches[1].globalY);
_distance = Point.distance(_p1, _p2);
} else if (_touches.length == 1)
_interactSystem.onInteraction(MouseEvent.MOUSE_DOWN, touch.globalX, touch.globalY);
break;
case TouchPhase.HOVER:
if
```

```
(_notifyOnMove && _touches.length == 1)
_interactSystem.onInteraction(MouseEvent.MOUSE_MOVE, touch.globalX, touch.globalY);
break;
case TouchPhase.MOVED:
if (_touches.length == 2)
{
_p1.setTo(_touches[0].globalX, _touches[0].globalY);
_p2.setTo(_touches[1].globalX, _touches[1].globalY);
var newDis:Number = Point.distance(_p1, _p2);
if (Math.abs(newDis - _distance) >= 35 * DeviceMetrics.DENSITY)
{
_zooming = true;
_interactSystem.onZoom(newDis > _distance ? .1 : -.1, touch.globalX, touch.globalY);
_distance = newDis;
}
} else if (_touches.length == 1)
_interactSystem.onInteraction(MouseEvent.MOUSE_MOVE, touch.globalX, touch.globalY);
break;
case TouchPhase.ENDED:
if (_touches.length == 4)
_interactSystem.onKey(KeyboardKey.Q.keyCode, true);
else if (_touches.length == 3)
{
if (touch.globalX > DeviceMetrics.WIDTH_PIXELS * .5)
_interactSystem.onKey(KeyboardKey.W.keyCode, true);
else
_interactSystem.onKey(KeyboardKey.S.keyCode, true);
} else if (_touches.length == 2 && !_zooming)
{
if (touch.globalX > DeviceMetrics.WIDTH_PIXELS * .5)
_interactSystem.onKey(KeyboardKey.D.keyCode, true);
else
_interactSystem.onKey(KeyboardKey.A.keyCode, true);
} else
_interactSystem.onInteraction(MouseEvent.MOUSE_UP, touch.globalX, touch.globalY);
break;
}
}

protected function onMouse( e:MouseEvent ):void
{
switch (e.type)
{
case MouseEvent.MOUSE_DOWN:
case MouseEvent.RIGHT_MOUSE_DOWN:
Application.STAGE.addEventListener(MouseEvent.MOUSE_MOVE, onMouse);
Application.STAGE.addEventListener(MouseEvent.MOUSE_UP, onMouse);
_interactSystem.onInteraction(MouseEvent.MOUSE_DOWN, e.stageX, e.stageY);
break;
case MouseEvent.MOUSE_MOVE:
_interactSystem.onInteraction(MouseEvent.MOUSE_MOVE,
```

```
e.stageX, e.stageY);
break;
case MouseEvent.MOUSE_UP:
case MouseEvent.RIGHT_MOUSE_UP:
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_UP, onMouse);
_interactSystem.onInteraction(MouseEvent.MOUSE_UP, e.stageX, e.stageY);
break;
}
}

protected function onMouseWheel( e:MouseEvent ):void { _interactSystem.onZoom(e.delta > 0
? .1 : -.1, e.stageX, e.stageY); }

protected function addListeners():void
{
if (Application.STARLING_ENABLED)
_layer.addEventListener(TouchEvent.TOUCH, onTouch);
else
{
_layer.addEventListener(MouseEvent.MOUSE_DOWN, onMouse);
_layer.addEventListener(MouseEvent.MOUSE_UP, onMouse);
}
Application.STAGE.addEventListener(MouseEvent.RIGHT_MOUSE_DOWN, onMouse);
Application.STAGE.addEventListener(MouseEvent.RIGHT_MOUSE_UP, onMouse);
Application.STAGE.addEventListener(MouseEvent.MOUSE_WHEEL, onMouseWheel);
}

protected function removeListeners():void
{
if (Application.STARLING_ENABLED)
_layer.removeEventListener(TouchEvent.TOUCH, onTouch);
else
{
_layer.removeEventListener(MouseEvent.MOUSE_DOWN, onMouse);
_layer.removeEventListener(MouseEvent.MOUSE_UP, onMouse);
}

Application.STAGE.removeEventListener(MouseEvent.RIGHT_MOUSE_DOWN, onMouse);
Application.STAGE.removeEventListener(MouseEvent.RIGHT_MOUSE_UP, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_UP, onMouse);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_WHEEL, onMouseWheel);
}

public function set notifyOnMove( v:Boolean ):void
{
_notifyOnMove = v;
if (!Application.STARLING_ENABLED)
{
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE,
```

```
onMouse);
if (v)
Application.STAGE.addEventListener(MouseEvent.MOUSE_MOVE, onMouse);
}
}

public function destroy():void
{
if (_state == StateEvent.GAME_BATTLE_INIT || _state == StateEvent.GAME_BATTLE || _state
== StateEvent.GAME_BATTLE_CLEANUP)
{
removeUpKey(KeyboardKey.ONE.keyCode);
removeUpKey(KeyboardKey.TWO.keyCode);
removeUpKey(KeyboardKey.THREE.keyCode);
removeUpKey(KeyboardKey.FOUR.keyCode);
removeUpKey(KeyboardKey.FIVE.keyCode);
removeUpKey(KeyboardKey.SIX.keyCode);
removeUpKey(KeyboardKey.SEVEN.keyCode);
removeUpKey(KeyboardKey.EIGHT.keyCode);
removeUpKey(KeyboardKey.NINE.keyCode);
removeUpKey(KeyboardKey.ZERO.keyCode);

removeUpKey(KeyboardKey.F1.keyCode);
removeUpKey(KeyboardKey.F2.keyCode);
removeUpKey(KeyboardKey.F3.keyCode);
removeUpKey(KeyboardKey.F4.keyCode);
removeUpKey(KeyboardKey.F5.keyCode);
removeUpKey(KeyboardKey.F6.keyCode);

removeUpKey(KeyboardKey.A.keyCode);
removeUpKey(KeyboardKey.D.keyCode);
removeUpKey(KeyboardKey.DOWN.keyCode);
removeUpKey(KeyboardKey.LEFT.keyCode);
removeUpKey(KeyboardKey.RIGHT.keyCode);
removeUpKey(KeyboardKey.Q.keyCode);
removeUpKey(KeyboardKey.S.keyCode);
removeUpKey(KeyboardKey.UP.keyCode);
removeUpKey(KeyboardKey.W.keyCode);
removeUpKey(KeyboardKey.F.keyCode);
removeUpKey(KeyboardKey.G.keyCode);
removeUpKey(KeyboardKey.T.keyCode);
removeUpKey(KeyboardKey.R.keyCode);

removeUpKey(KeyboardKey.ESCAPE.keyCode);
removeDownKey(KeyboardKey.CONTROL.keyCode);
removeUpKey(KeyboardKey.CONTROL.keyCode);
removeDownKey(KeyboardKey.SHIFT.keyCode);
removeUpKey(KeyboardKey.SHIFT.keyCode);
} else if (_state == StateEvent.GAME_SECTOR_INIT || _state == StateEvent.GAME_SECTOR
|| _state == StateEvent.GAME_SECTOR_CLEANUP)
{
```

```
removeUpKey(KeyboardKey.A.keyCode);
removeUpKey(KeyboardKey.D.keyCode);
removeUpKey(KeyboardKey.LEFT.keyCode);
removeUpKey(KeyboardKey.RIGHT.keyCode);
} else
{
removeDownKey(KeyboardKey.SHIFT.keyCode);
removeUpKey(KeyboardKey.SHIFT.keyCode);
removeUpKey(KeyboardKey.ESCAPE.keyCode);
}

removeListeners();
_interactSystem = null;
_keyboardController = null;
_layer = null;
_notifyOnMove = false;
_p1 = _p2 = null;
_touches.length = 0;
_touches = null;
}
}
}


-------------------------------------
File 266: igw\com\game\entity\systems\interact\controls\SelectorEntity.as
package com.game.entity.systems.interact.controls
{
import com.game.entity.factory.IInteractFactory;

import flash.utils.Dictionary;

import org.ash.core.Entity;
import org.shared.ObjectPool;

public class SelectorEntity
{
private static const LOOKUP:Dictionary = new Dictionary();

public static function getSelectorEntity( id:String ):SelectorEntity
{
if (LOOKUP.hasOwnProperty(id))
return LOOKUP[id];
return null;
}

private var _count:int = 0;
private var _entity:Entity;
private var _id:String;
private
```

```
var _interactFactory:IInteractFactory;

public function init( id:String, entity:Entity, factory:IInteractFactory ):void
{
_entity = entity;
if (_entity == null)
throw new Error("WTF! Man!");
_id = id;
_interactFactory = factory;
LOOKUP[_id] = this;
}

public function increaseCount():void { _count++; }
public function decreaseCount():void
{
_count--;
if (_count <= 0)
{
_interactFactory.destroyInteractEntity(_entity);
_entity = null;
ObjectPool.give(this);
}
}

public function get id():String { return _id; }

public function destroy():void
{
delete LOOKUP[_id];
_count = 0;
_entity = null;
_id = null;
_interactFactory = null;
}
}
}

-------------------------------------
File 267: igw\com\game\entity\systems\sector\FleetSystem.as
package com.game.entity.systems.sector
{
import com.Application;
import com.event.StateEvent;
import com.game.entity.components.shared.Detail;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.sector.fleet.FleetNode;
import com.game.entity.nodes.sector.fleet.FleetStarlingNode;
import com.game.entity.nodes.sector.fleet.IFleetNode;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import
```

```
com.util.BattleUtils;

import org.adobe.utils.StringUtil;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class FleetSystem extends System
{
private static const THRUSTER_THRESHOLD:Number = 5;

[Inject(nodeType="IFleetNode")]
public var nodes:NodeList;

private var _tempRotation:Number;
private var _prototypeModel:PrototypeModel;
private var _vfxFactory:IVFXFactory;

override public function addToGame( game:Game ):void
{
var fleetNode:FleetNode;
var fleetStarlingNode:FleetStarlingNode;

nodes.nodeAdded.add(onNodeAdded);
nodes.nodeRemoved.add(onNodeRemoved);
}

override public function update( time:Number ):void
{
for (var node:IFleetNode = nodes.head; node; node = node.inext)
{
if (node.move.totalTime >= THRUSTER_THRESHOLD && node.move.time > 1.5 &&
node.move.time < node.move.totalTime - 0.3)
{
if (!node.fleet.thrustersEngaged)
createThrusters(node);
} else if (node.fleet.thrustersEngaged)
destroyThrusters(node);

//update the rotation of the ship if it is moving
if (node.move.moving)
{
// Determine sprite frame to use based on angle
_tempRotation = Math.atan2(Math.sin(node.position.rotation) * 2.0,
Math.cos(node.position.rotation));
node.position.rotation = _tempRotation;
_tempRotation = (_tempRotation / Math.PI) * 180;
_tempRotation = _tempRotation % 360;
if (_tempRotation < 0)
_tempRotation += 360;
node.animation.label
```

```
= node.detail.spriteName + "_" + Math.round(_tempRotation / 3.025);
}

//update thrusters if they are engaged
if (node.fleet.thrustersEngaged)
{
BattleUtils.instance.moveToAttachPoint(node.ientity, node.fleet.thrusterBackLeft);
BattleUtils.instance.moveToAttachPoint(node.ientity, node.fleet.thrusterBackRight);
}
}
}

private function onNodeAdded( node:IFleetNode ):void
{
// Correct rotation for isometric perspective. only need to do this if not in battle
if (Application.STATE != StateEvent.GAME_BATTLE)
{
var x:Number = Math.cos(node.position.rotation);
var y:Number = Math.sin(node.position.rotation) * 2;
node.position.rotation = Math.atan2(y, x);
}

// Determine sprite frame to use based on angle
var rot:Number = (node.position.rotation / Math.PI) * 180;
rot = rot % 360;
if (rot < 0)
rot += 360;
var num:int = rot / 3.025 | 0;
node.animation.label = node.detail.spriteName + "_" + num;
}

private function createThrusters( node:IFleetNode ):void
{
// Show the thrusters
var attachPoints:Array = node.detail.prototypeVO.getValue("attachPoints");

for each (var attachPoint:String in attachPoints)
{
var attachPointProto:IPrototype = _prototypeModel.getAttachPoint(attachPoint);

// Ignore everything but back thrusters
if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"), "ThrusterBackwardLeft"))
node.fleet.thrusterBackLeft = _vfxFactory.createThruster(node.ientity, attachPointProto, false,
true);
if (StringUtil.beginsWith(attachPointProto.getValue("attachPointType"),
"ThrusterBackwardRight"))
node.fleet.thrusterBackRight = _vfxFactory.createThruster(node.ientity, attachPointProto, false,
true);
}
node.fleet.thrustersEngaged = true;
}
```

```
private function destroyThrusters( node:IFleetNode ):void
{
if (node.fleet.thrusterBackLeft && node.fleet.thrusterBackLeft.has(Detail))
_vfxFactory.destroyVFX(node.fleet.thrusterBackLeft);
if (node.fleet.thrusterBackRight && node.fleet.thrusterBackRight.has(Detail))
_vfxFactory.destroyVFX(node.fleet.thrusterBackRight);
node.fleet.disengageThrusters();
}

private function onNodeRemoved( node:IFleetNode ):void
{
if (node.fleet.thrustersEngaged)
destroyThrusters(node);
}

override public function removeFromGame( game:Game ):void
{
nodes.nodeAdded.remove(onNodeAdded);
nodes.nodeRemoved.remove(onNodeRemoved);
nodes = null;
}

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }
}
}


-----------------------------------
File 268: igw\com\game\entity\systems\shared\AnimationSystem.as
package com.game.entity.systems.shared
{
import com.enum.CategoryEnum;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.shared.AnimationNode;
import com.model.asset.AssetModel;
import com.model.asset.ISpritePack;

import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class AnimationSystem extends System
{
```

```
[Inject(nodeType="com.game.entity.nodes.shared.AnimationNode")]
public var nodes:NodeList;

private var _assetModel:AssetModel;
private var _vfxFactory:IVFXFactory;

override public function update( time:Number ):void
{
var node:AnimationNode;
var animation:Animation;
var detail:Detail;
var spritePack:ISpritePack;
var timeDiff:int;
var animationFrames:Array;

for (node = nodes.head; node; node = node.next)
{
detail = node.detail;
animation = node.animation;

if (animation.render && animation.render.alpha != animation.alpha)
animation.render.alpha = animation.alpha;

if (!animation.ready)
{
spritePack = _assetModel.getSpritePack(animation.type, true, node.entity);
if (!spritePack || !spritePack.ready)
continue;
animation.spritePack = spritePack;
animation.sprite = spritePack.getFrame(animation.label, animation.frame);
var animationFrames:Array = spritePack.getFrames(animation.label);
if(animationFrames)
animation.numberOfFrames = animationFrames.length;
else
animation.numberOfFrames = 0;

animation.labelChanged = false;
if (animation.render)
{
animation.render.updateFrame(animation.sprite, animation, true);
if (animation.center)
{
if (node.entity.has(Position))
Position(node.entity.get(Position)).dirty = true;
else
{
if (!animation.lostContext)
{
animation.render.x -= animation.offsetX;
animation.render.y
```

```
-= animation.offsetY;
}
animation.forceReady(true);
}
}
}
animation.dispatch(Animation.ANIMATION_READY);
} else if (animation.labelChanged)
{
animation.frame = 0;
animation.labelChanged = false;
animation.sprite = animation.spritePack.getFrame(animation.label, animation.frame);
animation.numberOfFrames = animation.spritePack.getFrames(animation.label).length;
if (animation.render)
animation.render.updateFrame(animation.sprite, animation, true);
animation.playing = true;
} else if (animation.textChanged && animation.text && animation.ready && animation.render)
{
if (animation.textLostContext)
{
Object(animation.render).text = null;
animation.textLostContext = false;
}
animation.render.color = animation.color;
Object(animation.render).text = animation.text;
animation.textChanged = false;
} else if (time != 0 && animation.numberOfFrames > 1 && animation.playing)
{
if (animation.randomStart)
{
animation.frame = (animation.numberOfFrames * Math.random());
animation.randomStart = false;
}

animation.time += time;
if (animation.time >= animation.frameDuration)
{
timeDiff = animation.time / animation.frameDuration | 0;
animation.frame += timeDiff;
if (animation.frame >= animation.numberOfFrames)
{
animation.frame = 0;
animation.dispatch(Animation.ANIMATION_COMPLETE);
if (!animation.replay)
animation.playing = false;
if (animation.destroyOnComplete)
{
if (detail.category == CategoryEnum.EXPLOSION)
_vfxFactory.destroyVFX(node.entity);
}
}
```

```
if (animation.spritePack)
{
animation.time -= timeDiff * animation.frameDuration;
animation.sprite = animation.spritePack.getFrame(animation.label, animation.frame);
if (animation.render)
animation.render.updateFrame(animation.sprite, animation);
}
}
}
}
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }

override public function removeFromGame( game:Game ):void
{
nodes = null;
_assetModel = null;
_vfxFactory = null;
}
}
}

-------------------------------------
File 269: igw\com\game\entity\systems\shared\FSMSystem.as
package com.game.entity.systems.shared
{
import com.enum.CategoryEnum;
import com.enum.TypeEnum;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.nodes.shared.FSMNode;

import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class FSMSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.shared.FSMNode")]
public var nodes:NodeList;

private var _starbaseFactory:IStarbaseFactory;

override public function addToGame( game:Game ):void
{
super.addToGame(game);
}
```

```
override public function update( time:Number ):void
{
for (var node:FSMNode = nodes.head; node; node = node.next)
{
if (!node.fsm.advanceState(node))
destroyNode(node);
}
}

private function destroyNode( node:FSMNode ):void
{
switch (node.detail.category)
{
case CategoryEnum.BUILDING:
_starbaseFactory.destroyStarbaseItem(node.entity);
break;
}
}

[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }

override public function removeFromGame( game:Game ):void
{
super.removeFromGame(game);
nodes = null;
}
}
}

-----------------------------------
File 270: igw\com\game\entity\systems\shared\MoveSystem.as
package com.game.entity.systems.shared
{
import com.Application;
import com.controller.ServerController;
import com.enum.CategoryEnum;
import com.enum.EntityMoveEnum;
import com.event.StateEvent;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.factory.IAttackFactory;
import com.game.entity.factory.IVFXFactory;
import com.game.entity.nodes.shared.MoveNode;
import com.model.fleet.FleetModel;

import
```

```
org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class MoveSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.shared.MoveNode")]
public var nodes:NodeList;

private var _fleetModel:FleetModel;
private var _tempRotation:Number;
private var _vfxFactory:IVFXFactory;
private var _attackFactory:IAttackFactory;

override public function update( time:Number ):void
{
if (time == 0)
return;

var node:MoveNode;
for (node = nodes.head; node; node = node.next)
{
moveEntity(node, time);
}
}

private function moveEntity( node:MoveNode, time:Number ):void
{
var detail:Detail = node.detail;
var move:Move = node.move;
var position:Position = node.position;

if (move.type == EntityMoveEnum.LERPING && !move.lerping && move.hasUpdate)
move.setNextUpdate();
if (move.directionChanged)
{
move.setStart(position.position);
if (move.type == EntityMoveEnum.POINT_TO_POINT)
move.setDelta(move.destination.x - move.start.x, move.destination.y - move.start.y);
else
move.setDelta(move.lerpDestination.x - move.start.x, move.lerpDestination.y - move.start.y);
move.directionChanged = false;
if (move.type == EntityMoveEnum.LERPING)
position.targetRotation = move.rotation;
else if (detail.category == CategoryEnum.SHIP)
{
position.targetRotation = Math.atan2(move.delta.y, move.delta.x);
} else
position.targetRotation = position.rotation = Math.atan2(move.delta.y, move.delta.x);
}
if
```

```
(move.moving)
{
if (move.startTick > ServerController.SIMULATED_TICK)
return;
move.time += time;
if (move.time < 0)
return;

//rotation
var ratio:Number = Math.min(move.totalTime, move.time) / move.totalTime;
//alpha fade out on projectiles that are reaching the end of their life
if (move.fadeOut > 0 && detail.category == CategoryEnum.ATTACK)
{
if (ServerController.SIMULATED_TICK >= move.fadeOut)
node.animation.alpha = 1.05 - ratio;
}
if (position.rotation != position.targetRotation)
{
if (move.type == EntityMoveEnum.LERPING)
position.rotation = position.startRotation + position.rotationDelta * ratio;
else
position.rotation = position.startRotation + Math.min(move.time * .6, 1) * position.rotationDelta;
}

if (move.relative)
{
move.setStart(position.linkedTo.position);
move.setDelta(move.destination.x, move.destination.y);
}

position.x = move.start.x + ratio * move.delta.x;
position.y = move.start.y + ratio * move.delta.y;
if (move.time >= move.totalTime)
{
if (move.type == EntityMoveEnum.POINT_TO_POINT)
{
move.moving = false;
if (Application.STATE == StateEvent.GAME_SECTOR && node.entity.has(Owned))
_fleetModel.updateFleet(null);

if (move.destroyOnComplete)
{
if (detail.category == CategoryEnum.ATTACK)
_attackFactory.destroyAttack(node.entity);
}
} else
{
if (move.hasUpdate)
{
var tt:Number = move.time - move.totalTime;
move.setNextUpdate();
```

```
move.time = tt;
moveEntity(node, 0);
} else
{
if (position.x == move.destination.x && position.y == move.destination.y)
move.moving = false;
move.lerping = false;
}
}
}
}
}

[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set vfxFactory( v:IVFXFactory ):void { _vfxFactory = v; }
[Inject]
public function set attackFactory( v:IAttackFactory ):void { _attackFactory = v; }

override public function removeFromGame( game:Game ):void
{
nodes = null;
_fleetModel = null;
_vfxFactory = null;
}
}
}

-----------------------------------
File 271: igw\com\game\entity\systems\shared\TweenSystem.as
package com.game.entity.systems.shared
{
import org.ash.core.Game;
import org.ash.core.System;
import org.greensock.TweenManual;

public class TweenSystem extends System
{
override public function update( time:Number ):void
{
TweenManual.updateAll(time);
}

override public function removeFromGame( game:Game ):void
{
TweenManual.killAll();
}
}
}
```

File 272: igw\com\game\entity\systems\shared\VCSystem.as
```
package com.game.entity.systems.shared
{
import com.enum.CategoryEnum;
import com.enum.TypeEnum;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.VCList;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.State;
import com.game.entity.factory.IVCFactory;
import com.game.entity.nodes.shared.visualComponent.IVCNode;
import com.game.entity.nodes.shared.visualComponent.VCNode;
import com.game.entity.nodes.shared.visualComponent.VCSpriteNode;
import com.game.entity.nodes.shared.visualComponent.VCSpriteStarlingNode;
import com.game.entity.nodes.shared.visualComponent.VCStarlingNode;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.service.language.Localization;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

/**
* Certain entities will have a 'VCList' (aka Visual Component List) component. A VCList contains the names
* of additional entities that we want to display when the entity becomes visible (ie. it is given a Render).
* When the entity is no longer visible then the additional entities are also removed. This is useful for showing
* health bars, shields, turrets, building animations, etc.
*/
public class VCSystem extends System
{
[Inject(nodeType="IVCNode")]
public var nodes:NodeList;

[Inject(nodeType="IVCSpriteNode")]
public var spriteNodes:NodeList;

private var _game:Game;
private var _playerModel:PlayerModel;
private var _vcFactory:IVCFactory;

override public function addToGame( game:Game ):void
{
```

```
var vcNode:VCNode;
var vcsNode:VCStarlingNode;
var vcSpriteNode:VCSpriteNode;
var vcSpriteStarlingNode:VCSpriteStarlingNode;
_game = game;
nodes.nodeAdded.add(onNodeAdded);
nodes.nodeRemoved.add(onNodeRemoved);
spriteNodes.nodeAdded.add(onNodeAdded);
spriteNodes.nodeRemoved.add(onNodeRemoved);
}

private function onNodeAdded( node:IVCNode ):void
{
var vcList:VCList = node.vcList;
//if there are components already created for this vcList then we want to ignore this step
if (vcList.components.length > 0)
return;
vcList.addCallbacks(addComponentToNode, rebuildComponent, removeComponentFromNode,
node.ientity);
var names:Array = vcList.names;
for (var i:int = 0; i < names.length; i++)
{
createComponentFromType(vcList, names[i], node.ientity);
}
}

private function addComponentToNode( entity:Entity, type:String ):void
{
var vcList:VCList = entity.get(VCList);
//only create the component if this class has been initialized
if ((nodes.head || spriteNodes.head) && Animation(entity.get(Animation)).render != null)
createComponentFromType(vcList, type, entity);
}

private function rebuildComponent( entity:Entity, type:String ):void
{
removeComponentFromNode(entity, type);
addComponentToNode(entity, type);
}

private function removeComponentFromNode( entity:Entity, type:String ):void
{
var vcList:VCList = entity.get(VCList);
for (var j:int = 0; j < vcList.components.length; j++)
{
if (Detail(vcList.components[j].get(Detail)).type == type)
{
switch (type)
{
default:
```

```
_vcFactory.destroyComponent(vcList.components[j]);
break;
}
vcList.components.splice(j, 1);
}
}
}

private function createComponentFromType( vcList:VCList, type:String, entity:Entity ):void
{
var animation:Animation;
var component:Entity;
switch (type)
{
case TypeEnum.BUILDING_ANIMATION:
component = _vcFactory.createBuildingAnimation(Building(entity.get(Building)).buildingVO,
entity);
if (component)
vcList.addComponent(component);
break;
case TypeEnum.BUILDING_CONSTRUCTION:

vcList.addComponent(_vcFactory.createBuildingConstruction(Building(entity.get(Building)).buildingVO,
entity));
break;
case TypeEnum.RESOURCE_DEPOT_CANISTER:
vcList.addComponent(_vcFactory.createDepotCannisters(entity));
break;
case TypeEnum.HEALTH_BAR:
vcList.addComponent(_vcFactory.createHealthBar(entity));
break;
case TypeEnum.DEBUFF_TRAY:
vcList.addComponent(_vcFactory.createDebuffTray(entity));
break;
case TypeEnum.ISO_1x1_IGA:
case TypeEnum.ISO_1x1_SOVEREIGNTY:
case TypeEnum.ISO_1x1_TYRANNAR:
case TypeEnum.ISO_2x2_IGA:
case TypeEnum.ISO_2x2_SOVEREIGNTY:
case TypeEnum.ISO_2x2_TYRANNAR:
case TypeEnum.ISO_3x3_IGA:
case TypeEnum.ISO_3x3_SOVEREIGNTY:
case TypeEnum.ISO_3x3_TYRANNAR:
vcList.addComponent(_vcFactory.createIsoSquare(entity, type));
break;
case TypeEnum.NAME:
var detail:Detail = entity.get(Detail);
var name:String = '';
var vo:PlayerVO = _playerModel.getPlayer(detail.ownerID);
if
```

```
(vo)
{
var locName:String = vo.name;
if (locName.indexOf('NPC.') != -1)
locName = Localization.instance.getString(vo.name);

if (detail.category != CategoryEnum.SHIP)
{
var level:int;
if (vo.isNPC)
level = detail.baseLevel;
else
level = vo.level;

name = "(" + level + ") " + locName;
} else
{
name = ((vo.isNPC) ? "(" + detail.level + ") " : "") + locName;
}
if(vo.isNPC)
{
if(detail.maxPlayersPerFaction == 1)
name += "\n(Solo Target)";
else if( detail.maxPlayersPerFaction > 1)
name += "\n(Max " + detail.maxPlayersPerFaction + ")";

}
else
name += ((vo.allianceName != '') ? '\n(' + vo.allianceName + ')' : '');

}

component = _vcFactory.createName(entity, name);
animation = component.get(Animation);
if (detail.category == CategoryEnum.SECTOR)
{
if (detail.type == TypeEnum.STARBASE_SECTOR_IGA ||
detail.type == TypeEnum.STARBASE_SECTOR_SOVEREIGNTY ||
detail.type == TypeEnum.STARBASE_SECTOR_TYRANNAR)
animation.offsetY = Animation(entity.get(Animation)).render.height * .50;
else
animation.offsetY = Animation(entity.get(Animation)).render.height * .60;
} else
animation.offsetY = Animation(entity.get(Animation)).render.height * .29;
vcList.addComponent(component);
break;
case TypeEnum.BUILDING_SHIELD:
case TypeEnum.TURRET_SHIELD:
component = _vcFactory.createBuildingShield(Building(entity.get(Building)).buildingVO, entity);
if (component)
vcList.addComponent(component);
```

```
break;
case TypeEnum.SHIELD:
vcList.addComponent(_vcFactory.createShield(entity));
break;
case TypeEnum.STATE_BAR:
var state:State = entity.get(State);
component = _vcFactory.createStateBar(entity, state.text);
Animation(component.get(Animation)).scaleX = State(entity.get(State)).percentageDone;
vcList.addComponent(component);
break;
case TypeEnum.STARBASE_TURRET:
component = _vcFactory.createTurret(Building(entity.get(Building)).buildingVO, entity);
if (component)
vcList.addComponent(component);
break;
case TypeEnum.STARBASE_SHIELD_IGA:
case TypeEnum.STARBASE_SHIELD_SOVEREIGNTY:
case TypeEnum.STARBASE_SHIELD_TYRANNAR:
vcList.addComponent(_vcFactory.createStarbaseShield(entity));
break;
}
}

private function onNodeRemoved( node:IVCNode ):void
{
var vcList:VCList = node.vcList;
var components:Vector.<Entity> = vcList.components;
for (var i:int = 0; i < components.length; i++)
{
_vcFactory.destroyComponent(components[i]);
}
vcList.components.length = 0;
vcList.removeCallbacks();
}

[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set vcFactory( v:IVCFactory ):void { _vcFactory = v; }

override public function removeFromGame( game:Game ):void
{
_game = null;
nodes.nodeAdded.remove(onNodeAdded);
nodes.nodeRemoved.remove(onNodeRemoved);
nodes = null;
spriteNodes.nodeAdded.remove(onNodeAdded);
spriteNodes.nodeRemoved.remove(onNodeRemoved);
spriteNodes = null;
_playerModel
```

```
= null;
_vcFactory = null;
}
}
}

------------------------------------
File 273: igw\com\game\entity\systems\shared\background\BackgroundItem.as
package com.game.entity.systems.shared.background
{
import com.util.rtree.RRectangle;

public class BackgroundItem
{
public var id:String;
public var bounds:RRectangle;
public var label:String;
public var layer:int;
public var width:Number;
public var height:Number;
public var parallaxSpeed:Number;
public var scale:Number;
public var type:String;
public var x:Number;
public var y:Number;

public function BackgroundItem( id:String, type:String, label:String, layer:int,
parallaxSpeed:Number, x:Number, y:Number, width:Number, height:Number, scale:Number )
{
this.id = id;
this.label = label;
this.layer = layer;
this.width = width * scale;
this.height = height * scale;
this.parallaxSpeed = parallaxSpeed;
this.scale = scale;
this.type = type;
this.x = x;
this.y = y
}
}
}

------------------------------------
File 274: igw\com\game\entity\systems\shared\background\BackgroundLayer.as
package com.game.entity.systems.shared.background
{
import com.util.rtree.RRectangle;
import com.util.rtree.RTree;

import
```

```
flash.geom.Rectangle;
import flash.utils.Dictionary;

import org.parade.util.DeviceMetrics;

public class BackgroundLayer
{
private var _bounds:Rectangle;
private var _id:int;
private var _layer:int;
private var _lookup:Dictionary;
private var _parallaxSpeed:Number;
private var _rect:RRectangle;
private var _scale:Number;
private var _tree:RTree;

public function init( layer:int, parallaxSpeed:Number, scale:Number, width:Number,
height:Number, lookup:Dictionary ):void
{
_id = 10000 * layer;
_layer = layer;
_lookup = lookup;
_parallaxSpeed = parallaxSpeed;
_rect = new RRectangle(0, 0, 0, 0);
_scale = scale;
_tree = new RTree();
_bounds = new Rectangle(0, 0, DeviceMetrics.MAX_WIDTH_PIXELS + width * _parallaxSpeed,
DeviceMetrics.MAX_HEIGHT_PIXELS + height * _parallaxSpeed);
}

public function addItemFromData( type:String, name:String, width:Number, height:Number,
scale:Number = 1, tile:Boolean = false, x:Number = 0, y:Number = 0, addToTree:Boolean = true
):BackgroundItem
{
var item:BackgroundItem;
var rect:RRectangle = new RRectangle(0, 0, 0, 0);

if (!tile)
{
item = new BackgroundItem(id, type, name, _layer, _parallaxSpeed, x, y, width, height, _scale *
scale);
rect = new RRectangle(item.x, item.y, item.x + item.width, item.y + item.height);
item.bounds = rect;
if (addToTree)
_tree.addRRectangle(rect, item.id);
_lookup[item.id] = item;
} else
{
scale = _scale * scale;
var dwidth:Number = 0;
var
```

```
dheight:Number = 0;
while (dheight <= _bounds.height)
{
while (dwidth <= _bounds.width)
{
item = new BackgroundItem(id, type, name, _layer, _parallaxSpeed, dwidth, dheight, width,
height, scale);
rect = new RRectangle(dwidth, dheight, dwidth + item.width, dheight + item.height);
item.bounds = rect;
if (addToTree)
_tree.addRRectangle(rect, item.id);
_lookup[item.id] = item;
dwidth += item.width;
}
dwidth = 0;
dheight += item.height;
}
}
return item;
}

public function addItem( item:BackgroundItem ):void
{
_tree.addRRectangle(item.bounds, item.id);
}

public function getItemsByRect( viewArea:Rectangle ):Array
{
_rect.setValues(viewArea.x * _parallaxSpeed, viewArea.y * _parallaxSpeed, viewArea.x *
_parallaxSpeed + viewArea.width, viewArea.y * _parallaxSpeed + viewArea.height);
return _tree.contains(_rect);
}

public function get bounds():Rectangle { return _bounds; }
public function get parallaxSpeed():Number { return _parallaxSpeed; }

private function get id():String { _id++; return "bg" + _id; }

public function destroy():void
{
_bounds = null;
_lookup = null;
_rect = null;
_tree = null;
}
}
}

-----------------------------------
File 275: igw\com\game\entity\systems\shared\background\BackgroundSystem.as
package
```

```
com.game.entity.systems.shared.background
{
import com.Application;
import com.enum.FactionEnum;
import com.enum.TypeEnum;
import com.event.StateEvent;
import com.event.signal.QuadrantSignal;
import com.game.entity.factory.IBackgroundFactory;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.asset.ISpritePack;
import com.model.battle.BattleModel;
import com.model.scene.SceneModel;
import com.model.sector.SectorModel;
import com.service.loading.LoadPriority;
import com.util.InteractEntityUtil;
import com.util.Random;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.geom.Rectangle;
import flash.utils.Dictionary;

import org.ash.core.Game;
import org.ash.core.System;
import org.osflash.signals.Signal;
import org.parade.core.IViewStack;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;
import org.starling.display.BlendMode;
import org.starling.display.Image;
import org.starling.display.Sprite;

public class BackgroundSystem extends System
{
private static const MAX_MOONS:int = 3; //Does a random number between 0 and
MAX_MOONS. Max is actually one less than the value set here
private static const MAX_PLANETS:int = 2; //Does a random number between 0 and
MAX_PLANETS. Max is actually one less than the value set here

private static const STAR_LAYER:int = 0;
private static const STAR_LAYER2:int = 1;
private static const PLANET_LAYER_1:int = 2;
private static const MOON_LAYER_1:int = 3;
private static const MOON_LAYER_2:int = 4;
private static const MOON_LAYER_3:int = 5;
private static const ASTEROID_LAYER_1:int = 6;
private static const ASTEROID_LAYER_2:int = 7;
private static const FOG_LAYER:int = 8;
private
```

```
static const DEBRIS_LAYER:int = 9;

[Inject]
public var assetModel:AssetModel;
[Inject]
public var backgroundFactory:IBackgroundFactory;
[Inject]
public var quadrantSignal:QuadrantSignal;
[Inject]
public var sceneModel:SceneModel;
[Inject]
public var sectorModel:SectorModel;
[Inject]
public var viewStack:IViewStack;

private var _game:Game;
private var _initialized:Boolean;
private var _layers:Vector.<BackgroundLayer>;
private var _lookup:Dictionary;
private var _nebula:flash.display.Sprite;
private var _nebulaStarling:org.starling.display.Sprite;
private var _oldIds:Array;
private var _oldView:Rectangle;
private var _readySignal:Signal;

override public function addToGame( game:Game ):void
{
_game = game;
_initialized = false;
_layers = new Vector.<BackgroundLayer>;
_lookup = new Dictionary();
_oldIds = [];
_oldView = new Rectangle();
_readySignal = new Signal();

quadrantSignal.add(onVisibleHashChanged);
}

override public function update( time:Number ):void
{
if (!_initialized && sceneModel.ready && assetModel.getSpritePack(TypeEnum.BACKGROUND,
true, null, LoadPriority.IMMEDIATE))
{
if (Application.STARLING_ENABLED)
createNebulaStarfield();
else
createNebula();
}
}

public
```

```
function buildBackground(battleModel:BattleModel, useModelData:Boolean = false):void
{
var assetVO:AssetVO;
var item:BackgroundItem;
var num:int;
var width:Number = (Application.STATE == StateEvent.GAME_SECTOR) ? sectorModel.width :
5000;
var height:Number = (Application.STATE == StateEvent.GAME_SECTOR) ? sectorModel.height
: 5000;

if(useModelData && Application.STATE == StateEvent.GAME_BATTLE &&
battleModel.mapSizeX > 0 && battleModel.mapSizeY > 0)
{
width = battleModel.mapSizeX;
height = battleModel.mapSizeY / 2;
}

var customAppearance:Boolean = false;

var spriteSheets:String;
var spriteSheetPath:String;
if(useModelData && battleModel.galacticName.length > 0)
{
customAppearance = true;
spriteSheetPath = "bg/" + battleModel.galacticName + "/";
}
else
{
//setup the spritesheet for the space elements
spriteSheetPath = "bg/IMP/";
if ( sectorModel.sectorFaction != FactionEnum.IMPERIUM)
{
if (sectorModel.sectorFaction == FactionEnum.IGA)
spriteSheetPath = "bg/IGA/";
else
spriteSheetPath = (sectorModel.sectorFaction == FactionEnum.SOVEREIGNTY) ? "bg/SOV/" :
"bg/TYR/";
}
}

var random:Random;
var seed:String;
if(customAppearance)
{
random = new Random(battleModel.appearanceSeed);
seed = String(battleModel.appearanceSeed);
spriteSheets = spriteSheetPath + "BG" + battleModel.backgroundId;
}
else
{
random
```

```
= new Random(sectorModel.appearanceSeed);
seed = String(sectorModel.appearanceSeed);
spriteSheets = spriteSheetPath + "BG" + random.nextMinMax(1,
sectorModel.numBackgroundSprites + 1);
}
createLayers(width, height);

//create the nebulae
updateSpriteData(TypeEnum.BACKGROUND, spriteSheets, true);

//create the starfields
spriteSheets = spriteSheetPath + "StarfieldFog";
updateSpriteData(TypeEnum.BACKGROUND_FOG_STARS, spriteSheets);
if (Application.STARLING_ENABLED) //don't show bg stars if in software mode for performance
_layers[STAR_LAYER].addItemFromData(TypeEnum.BACKGROUND_FOG_STARS, "StarBG",
2048, 1048, 1, true);
_layers[STAR_LAYER2].addItemFromData(TypeEnum.BACKGROUND_FOG_STARS,
"StarFG", 1024, 1024, 1, true);

spriteSheets = "";
//only show the planets, moons and asteroids if we're in the sector
//if (Application.STATE == StateEvent.GAME_SECTOR)
{
//create the planets
var numPlanets:int;

if(customAppearance)
{
if(battleModel.planetId == 0)
numPlanets = 0;
else
numPlanets = 1;
}
else
numPlanets = random.nextMinMax(0, MAX_PLANETS);
if (numPlanets == 0)
numPlanets = (random.nextNumber() < .92) ? 1 : 0;
if (numPlanets > 0)
{
var bounds:Rectangle;
var scale:Number;
var ss:int ;

if(customAppearance)
ss = battleModel.planetId;
else
ss = random.nextMinMax(1, sectorModel.numPlanetSprites + 1);
spriteSheets += spriteSheetPath + "Planet" + ss;
var planets:Vector.<BackgroundItem> = new Vector.<BackgroundItem>;
for (var i:int = 0; i < numPlanets; i++)
{
```

```
bounds = _layers[PLANET_LAYER_1].bounds;
scale = 1 + (random.nextMinMax(0, 35) / 100);
item =
_layers[PLANET_LAYER_1].addItemFromData(TypeEnum.BACKGROUND_ELEMENTS,
"Planet" + ss, 1024, 1024, scale, false,
random.nextMinMax(0 - (bounds.width * .15), bounds.width - (bounds.width * .3)),
random.nextMinMax(0 - (bounds.height * .15), bounds.height - (bounds.height * .3)));
planets.push(item);
}

//create the moons
var numMoons:int;
var parallax:Number;
for (i = 0; i < planets.length; i++)
{
item = planets[i];
if(customAppearance)
numMoons = battleModel.moonQuantity;
else
numMoons = random.nextMinMax(0, MAX_MOONS);

for (i = 0; i < numMoons; i++)
{
num = (i % 3) + MOON_LAYER_1;
parallax = 1 - _layers[num].parallaxSpeed;
_layers[num].addItemFromData(TypeEnum.BACKGROUND_ELEMENTS, "Moon" +
random.nextMinMax(1, 3), 256, 256, 1, false,
random.nextMinMax((item.x + (item.width * .13)) / parallax, (item.x + item.width - (item.width *
.13)) / parallax),
random.nextMinMax((item.y + (item.height * .13)) / parallax, (item.y + item.height - (item.height *
.13)) / parallax));
}

}
}

//create the asteroids
var numAsteroids:int;
if(customAppearance)
numAsteroids = battleModel.asteroidQuantity;
else
numAsteroids = random.nextMinMax(1, 17);
if (numAsteroids > 0 && Application.STARLING_ENABLED)
{
bounds = _layers[ASTEROID_LAYER_1].bounds;
var bounds2:Rectangle = _layers[ASTEROID_LAYER_2].bounds;
if (spriteSheets != "")
spriteSheets += ",";
spriteSheets += spriteSheetPath + "Asteroids";
var
```

```
p:Number = _layers[ASTEROID_LAYER_1].parallaxSpeed;
for (i = 0; i < numAsteroids; i++)
{
item =
_layers[ASTEROID_LAYER_2].addItemFromData(TypeEnum.BACKGROUND_ELEMENTS,
"AsteroidFG", 2048, 1024, 1, false,
random.nextMinMax(0 - (bounds2.width * .2), bounds2.width - (bounds2.width * .2)),
random.nextMinMax(0 - (bounds2.height * .2), bounds2.height - (bounds2.height * .2)));
//should we show the background asteroid?
if (random.nextNumber() < .7)
_layers[ASTEROID_LAYER_1].addItemFromData(TypeEnum.BACKGROUND_ELEMENTS,
"AsteroidBG", 2048, 1024, 1, false, item.x - item.x * p, item.y - item.y * p);

}
}

//create space debris

updateSpriteData(TypeEnum.BACKGROUND_ELEMENTS, spriteSheets);
}

//create space fog
_layers[FOG_LAYER].addItemFromData(TypeEnum.BACKGROUND_FOG_STARS, "Fog",
1024, 1024, 1, true);

sceneModel.buildScene(width, height);
}

private function updateSpriteData( type:String, spriteSheets:String, isJPG:Boolean = false ):void
{
var assetVO:AssetVO = assetModel.getEntityData(type);
var make:Boolean = false;
if (!assetVO)
make = true;
else if (assetVO && assetVO.spriteSheetsString != spriteSheets)
{
assetModel.removeGameAssetData(type);
var sp:ISpritePack = assetModel.getSpritePack(type, false);
assetModel.removeSpritePack(sp);
make = spriteSheets != "";
}
if (make)
assetModel.addGameAssetData(InteractEntityUtil.createPrototype(type, 8, spriteSheets,
isJPG));
}

private function onVisibleHashChanged( type:int, viewBounds:Rectangle ):void
{
if (type == QuadrantSignal.VISIBLE_HASH_CHANGED)
{
var
```

```
bgItem:BackgroundItem;
var duplicate:Dictionary = new Dictionary(true);
var ids:Array = getItemsByRect(viewBounds);
var index:int;
for (var i:int = 0; i < ids.length; i++)
{
if (!_game.getEntity(ids[i]))
{
backgroundFactory.createBackground(_lookup[ids[i]]);
}
duplicate[ids[i]] = true;
}
for (i = 0; i < _oldIds.length; i++)
{
if (!duplicate[_oldIds[i]])
backgroundFactory.destroyBackground(_game.getEntity(_oldIds[i]));
}
_oldIds = ids;
}
}

private function createNebula():void
{
var sp:ISpritePack = assetModel.getSpritePack(TypeEnum.BACKGROUND, true, null,
LoadPriority.IMMEDIATE);
if (sp && sp.ready)
{
var dwidth:Number = 0;
var dheight:Number = 0;
_nebula = new flash.display.Sprite();
var bmp:Bitmap;
while (dheight <= DeviceMetrics.MAX_HEIGHT_PIXELS)
{
while (dwidth <= DeviceMetrics.MAX_WIDTH_PIXELS)
{
bmp = new Bitmap(sp.getFrame("BG", 0));
bmp.x = dwidth;
bmp.y = dheight;
_nebula.addChild(bmp);
dwidth += 2048;
}
dwidth = 0;
dheight += 2048;
}
if (_nebula.numChildren == 1)
{
_nebula.x = (DeviceMetrics.MAX_WIDTH_PIXELS - _nebula.width) / 2;
_nebula.y = (DeviceMetrics.MAX_HEIGHT_PIXELS - _nebula.height) / 2;
}
_nebula.cacheAsBitmap = true;
viewStack.addToLayer(_nebula,
```

```
ViewEnum.GAME);
_initialized = true;
_readySignal.dispatch();
}
}

private function createNebulaStarfield():void
{
var sp:ISpritePack = assetModel.getSpritePack(TypeEnum.BACKGROUND, true, null,
LoadPriority.IMMEDIATE);
if (sp && sp.ready)
{
var dwidth:Number = 0;
var dheight:Number = 0;
_nebulaStarling = new org.starling.display.Sprite();
var image:Image;
while (dheight <= DeviceMetrics.MAX_HEIGHT_PIXELS)
{
while (dwidth <= DeviceMetrics.MAX_WIDTH_PIXELS)
{
image = new Image(sp.getFrame("BG", 0));
image.x = dwidth;
image.y = dheight;
image.blendMode = BlendMode.NONE;
_nebulaStarling.addChild(image);
dwidth += 2048;
}
dwidth = 0;
dheight += 2048;
}
if (_nebulaStarling.numChildren == 1)
{
_nebulaStarling.x = (DeviceMetrics.MAX_WIDTH_PIXELS - _nebulaStarling.width) / 2;
_nebulaStarling.y = (DeviceMetrics.MAX_HEIGHT_PIXELS - _nebulaStarling.height) / 2;
}
_nebulaStarling.blendMode = BlendMode.NONE;
_nebulaStarling.flatten();
viewStack.addToLayer(_nebulaStarling, ViewEnum.GAME);
_initialized = true;
_readySignal.dispatch();
}
}

private function getItemsByRect( viewBounds:Rectangle ):Array
{
var ids:Array = [];
for (var i:int = 0; i < _layers.length; i++)
{
ids = ids.concat(_layers[i].getItemsByRect(viewBounds));
}

return
```

```
ids;
}

public function addReadySignal( callback:Function ):void { _readySignal.addOnce(callback); }

/**
* Called when the stage3d context is lost to rebuild the starfield.
* Also called when this class is destroyed to allow the starfield to be garbage collected
*/
public function uninitialize():void
{
_initialized = false;
if (_nebula)
{
_nebula.parent.removeChild(_nebula);
_nebula = null;
} else if (_nebulaStarling)
{
_nebulaStarling.parent.removeChild(_nebulaStarling);
_nebulaStarling = null;
}
}

private function createLayers( width:Number, height:Number ):void
{
//starfield layers
_layers.push(createLayer(STAR_LAYER, .005, 1, width, height, _lookup));
_layers.push(createLayer(STAR_LAYER2, .009, 1, width, height, _lookup));

//planet layers
_layers.push(createLayer(PLANET_LAYER_1, .01, 1, width, height, _lookup));

//moon layers
_layers.push(createLayer(MOON_LAYER_1, .025, .50, width, height, _lookup));
_layers.push(createLayer(MOON_LAYER_2, .03, .75, width, height, _lookup));
_layers.push(createLayer(MOON_LAYER_3, .035, 1, width, height, _lookup));

//asteroid layers
_layers.push(createLayer(ASTEROID_LAYER_1, .13, 1.1, width, height, _lookup));
_layers.push(createLayer(ASTEROID_LAYER_2, .15, 1.1, width, height, _lookup));

//fog layers
_layers.push(createLayer(FOG_LAYER, .25, 1, width, height, _lookup));

//debris layer
_layers.push(createLayer(DEBRIS_LAYER, .35, 1, width, height, _lookup));
}

private function createLayer( layer:int, parallaxSpeed:Number, scale:Number, width:Number,
height:Number, lookup:Dictionary ):BackgroundLayer
{
```

```
var bgLayer:BackgroundLayer = ObjectPool.get(BackgroundLayer);
bgLayer.init(layer, parallaxSpeed, scale, width, height, lookup);
return bgLayer;
}

public function get ready():Boolean { return _initialized; }

override public function removeFromGame( game:Game ):void
{
uninitialize();
_game = null;
quadrantSignal.remove(onVisibleHashChanged);
quadrantSignal = null;
sceneModel.cleanup();
sceneModel = null;
backgroundFactory = null;
_readySignal.removeAll();
_readySignal = null;
viewStack = null;
_oldIds.length = 0;
_oldIds = null;
_oldView = null;

updateSpriteData(TypeEnum.BACKGROUND_ELEMENTS, "");

for (var i:int = 0; i < _layers.length; i++)
{
ObjectPool.give(_layers[i]);
}
_layers.length = 0;

sectorModel = null;
}
}
}

-------------------------------------
File 276: igw\com\game\entity\systems\shared\grid\GridField.as
package com.game.entity.systems.shared.grid
{
import flash.geom.Point;
import flash.geom.Rectangle;
import flash.utils.Dictionary;

public class GridField
{
public var CELL_SIZE:int;
public var COLUMNS:int;
public var CONVERSION:Number;
public
```

```
var ROWS:int;

private var _cells:Dictionary;
private var _columnSpan:Point;
private var _lookup:Dictionary;
private var _rowSpan:Point;
private var _testSpan:Point;

public function GridField( cellSize:int, bounds:Rectangle )
{
_cells = new Dictionary();
_columnSpan = new Point();
_lookup = new Dictionary();
_rowSpan = new Point();
_testSpan = new Point();

CELL_SIZE = cellSize;
CONVERSION = 1 / CELL_SIZE;
COLUMNS = Math.ceil(bounds.width / CELL_SIZE);
ROWS = Math.ceil(bounds.height / CELL_SIZE);
}

public function addToCell( object:*, hash:int ):void
{
if (hash < 0)
return;
//create a new cell if needed
if (!_cells.hasOwnProperty(hash))
createCell(hash);
//add the object to the cell
_cells[hash].push(object);
//add a lookup to the cells the object is in
if (!_lookup[object])
_lookup[object] = new Vector.<int>;
_lookup[object].push(hash);
}

public function removeFromCell( object:*, hash:int ):void
{
//remove the object
var index:int = _cells[hash].indexOf(object);
if (index != -1)
{
_cells[hash].splice(index, 1);
if (_cells[hash].length == 0)
{
_cells[hash] = null;
delete _cells[hash];
}
}
//remove
```

```
the hash
index = _lookup[object].indexOf(hash);
if (index != -1)
{
_lookup[object].splice(index, 1);
}
}

public function removeFromAllCells( object:* ):void
{
if (!_lookup[object])
return;
while (_lookup[object].length > 0)
{
removeFromCell(object, _lookup[object][0]);
}
_lookup[object] = null;
delete _lookup[object];
}

public function getHash( keyX:Number, keyY:Number ):int
{
return (keyX * CONVERSION | 0) + (keyY * CONVERSION | 0) * COLUMNS;
}

public function getKey( hash:int, point:Point ):void
{
point.x = (hash % COLUMNS) * CELL_SIZE;
point.y = (hash / COLUMNS | 0) * CELL_SIZE;
}

/**
* Returns the hash of a cell that is offset from a starting cell, respecting the bounds of the grid.
*
* @param hash The hash of the starting cell
* @param xOffset The number of cells to offset in x. Positive & negative offsets are honored.
* @param yOffset The number of cells to offset in y. Positive & negative offsets are honored.
* @return The hash of the cell that is offset from the starting cell, if within the grid bounds, or
* the cell at the edge of the grid bounds otherwise.
*/
public function getClampedCellByOffset( hash:int, xOffset:int, yOffset:int ):int
{
var result:int = hash;
result += xOffset;
var leftClamp:int = hash - hash % COLUMNS;
var rightClamp:int = leftClamp + COLUMNS - 1;
if (result < leftClamp)
result = leftClamp;
if (result > rightClamp)
result = rightClamp;

var
```

```
topClamp:int = result % COLUMNS;
var bottomClamp:int = (ROWS - 1) * COLUMNS + topClamp;
result += (yOffset * COLUMNS);
if (result < topClamp)
result = topClamp;
if (result > bottomClamp)
result = bottomClamp;

return result;
}

public function withinBounds( hash:int, boundMin:int, boundMax:int ):Boolean
{
_columnSpan.x = boundMin % COLUMNS;
_columnSpan.y = boundMax % COLUMNS;
_rowSpan.x = (boundMin / COLUMNS | 0);
_rowSpan.y = (boundMax / COLUMNS | 0);
_testSpan.x = hash % COLUMNS;
_testSpan.y = (hash / COLUMNS | 0);
if (_testSpan.x >= _columnSpan.x && _testSpan.x <= _columnSpan.y && _testSpan.y >=
_rowSpan.x && _testSpan.y <= _rowSpan.y)
return true;
return false;
}

public function getHashesInRect( rect:Rectangle ):Array
{
var result:Array = [];
var hashTL:int = getHash(rect.left, rect.top);
var hashTR:int = getHash(rect.right, rect.top);
var hashBR:int = getHash(rect.right, rect.bottom);
var x:int, y:int;
while (hashTL + x + y <= hashBR)
{
for (; hashTL + x <= hashTR; ++x)
{
result.push(hashTL + x + y);
trace("Quadrant cell: " + String(hashTL + x + y));
}

x = 0;
y += ROWS;
}

return result;
}

public function getObjects( hash:int ):Array
{
if (_cells.hasOwnProperty(hash))
return
```

```
_cells[hash];
return null;
}

private function createCell( hash:int ):void
{
if (hash < 0)
return;
if (_cells[hash])
return;
_cells[hash] = [];
}

private function removeCell( hash:int ):void
{
if (!_cells[hash])
return;
while (_cells[hash])
{
removeFromCell(_cells[hash][0], hash);
}
_cells[hash] = null;
delete _cells[hash];
}

public function get cells():Dictionary { return _cells; }

public function get lookup():Dictionary { return _lookup; }

public function destroy():void
{
for (var i:* in _cells)
{
removeCell(i);
}
_cells = null;
_columnSpan = null;
_rowSpan = null;
_testSpan = null;
_lookup = null;
}
}
}

-------------------------------------
File 277: igw\com\game\entity\systems\shared\grid\GridSystem.as
package com.game.entity.systems.shared.grid
{
import com.Application;
import com.controller.ServerController;
import
```

```
com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.StateEvent;
import com.event.signal.InteractSignal;
import com.event.signal.QuadrantSignal;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.Grid;
import com.game.entity.components.shared.Position;
import com.game.entity.nodes.shared.grid.GridMoveNode;
import com.game.entity.nodes.shared.grid.GridNode;
import com.game.entity.nodes.shared.grid.IGridNode;
import com.model.scene.SceneModel;
import com.presenter.sector.IMiniMapPresenter;
import com.service.server.outgoing.sector.SectorSetViewLocationRequest;

import flash.geom.Point;
import flash.geom.Rectangle;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;

public class GridSystem extends System
{
/** Number of grid cells to look in each direction. Not exactly a "radius," but close enough. */
private const QUADRANT_RADIUS:int = 3;

[Inject]
public var interactSignal:InteractSignal;
[Inject(nodeType="com.game.entity.nodes.shared.grid.GridNode")]
public var nodes:NodeList;
[Inject(nodeType="com.game.entity.nodes.shared.grid.GridMoveNode")]
public var movingNodes:NodeList;
[Inject]
public var quadrantSignal:QuadrantSignal;
[Inject]
public var sceneModel:SceneModel;

private var _game:Game;
private var _grid:GridField;
private var _miniMapPresenter:IMiniMapPresenter;
private var _newColumnSpan:Point;
private var _newRowSpan:Point;
private var _oldColumnSpan:Point;
private var _oldRowSpan:Point;
private var _point:Point;
private var _queue:Vector.<IGridNode>;

/** Tracks the hashes of the bounding cells of the quadrant. x = TL, y = BR */
private
```

```
var _quadrantHash:Point;

/** The actual coordinate bounds of the quadrant. */
private var _quadrantBounds:Rectangle;

private var _serverController:ServerController;
private var _testRect:Rectangle;
private var _visibleHash:Point;
private var _visibleBounds:Rectangle;

public function GridSystem() {}

override public function addToGame( game:Game ):void
{
_game = game;
_newColumnSpan = new Point();
_newRowSpan = new Point();
_oldColumnSpan = new Point();
_oldRowSpan = new Point();
_point = new Point();
_queue = new Vector.<IGridNode>;
_quadrantHash = new Point();
_quadrantBounds = new Rectangle(-100, -100, 1, 1);
_testRect = new Rectangle();
_visibleHash = new Point();
_visibleBounds = new Rectangle(-100, -100, 1, 1);
interactSignal.add(onScroll);
//add any entities that exist into their correct quadrants
for (var node:GridNode = nodes.head; node; node = node.next)
onEntityCreated(node);
for (var moveNode:GridMoveNode = movingNodes.head; moveNode; moveNode =
moveNode.next)
onEntityCreated(moveNode);

//listen for new entities
nodes.nodeAdded.add(onEntityCreated);
nodes.nodeRemoved.add(onEntityRemoved);
}

override public function update( time:Number ):void
{
if (!sceneModel.ready)
return;
//we must know the width and height of entities before adding them to quadrants
//check to see if any in the queue now have a width and height that we can use
if (_queue.length > 0)
{
for (var i:int = _queue.length - 1; i > -1; i--)
{
if (_queue[i].animation.width > 0)
{
```

```
onEntityCreated(_queue[i]);
_queue.splice(i, 1);
}
}
}

var newHashTL:int;
var newHashBR:int;
for (var node:GridMoveNode = movingNodes.head; node; node = node.next)
{
if (node.move.moving && node.animation.ready)
{
newHashTL = _grid.getHash(node.position.x - node.animation.offsetX, node.position.y -
node.animation.offsetY);
newHashBR = _grid.getHash(node.position.x - node.animation.offsetX + node.animation.width,
node.position.y - node.animation.offsetY + node.animation.height);
if (newHashTL != node.grid.hashTL || newHashBR != node.grid.hashBR)
{
onEntityRemoved(node);
onEntityCreated(node);
}
}
}
}

public function getEntitiesAt( x:Number, y:Number ):Array
{
var hash:int = _grid.getHash(x, y);
return _grid.getObjects(hash);
}

public function forceGridCheck( entity:Entity ):void
{
var grid:Grid = entity.get(Grid);
var position:Position = entity.get(Position);
var animation:Animation = entity.get(Animation);
if (!grid || !animation.ready)
return;
var newHashTL:int = _grid.getHash(position.x - animation.offsetX, position.y -
animation.offsetY);
var newHashBR:int = _grid.getHash(position.x - animation.offsetX + animation.width, position.y -
animation.offsetY + animation.height);
if (newHashTL != grid.hashTL || newHashBR != grid.hashBR)
{
for (var node:GridNode = nodes.head; node; node = node.next)
{
if (node.entity == entity)
{
onEntityRemoved(node);
onEntityCreated(node);
```

```
break;
}
}
}
}

/**
* A new entity has been created. Add it to the cells it belongs to
* @param node The entity that was created
*/
private function onEntityCreated( node:IGridNode ):void
{
if (node.animation.width > 0 && sceneModel.ready)
{
var hashTL:int = _grid.getHash(node.position.x - node.animation.offsetX, node.position.y -
node.animation.offsetY);
var hashBR:int = _grid.getHash(node.position.x - node.animation.offsetX +
node.animation.width,
node.position.y - node.animation.offsetY + node.animation.height);
node.grid.hashTL = hashTL;
node.grid.hashBR = hashBR;
var len:int = hashBR % _grid.COLUMNS;
var i:int = hashTL;
while (i <= hashBR)
{
_grid.addToCell(node, i);

//check to see if it is visible
if (!node.animation.visible && _grid.withinBounds(i, _visibleHash.x, _visibleHash.y))
node.animation.visible = true;
if (i % _grid.COLUMNS == len)
{
hashTL += _grid.COLUMNS;
i = hashTL;
} else
i++;
}

_miniMapPresenter.addToMiniMapSignal.dispatch(node.ientity, _quadrantBounds);
} else
{
//not ready so queue up or remove from the queue if needed
_queue.push(node);
}
}

/**
* A new entity has been destroyed. Remove it from the quadrants it belongs too
* @param node The entity that was destroyed
*/
```

```
private function onEntityRemoved( node:IGridNode ):void
{
if (node.animation.width > 0 && sceneModel.ready)
{
_grid.removeFromAllCells(node);
_miniMapPresenter.removeFromMiniMapSignal.dispatch(node.ientity);
} else
{
var index:int = _queue.indexOf(GridNode(node));
if (index > -1)
_queue.splice(index, 1);
}
}

/**
* Called whenever the player scrolls the screen
* @param type The type of signal being dispatched.
* @param dx The distance along the x axis that the screen scrolled
* @param dy The distance along the y axis that the screen scrolled
*/
private function onScroll( type:String, dx:Number, dy:Number ):void
{
if (!sceneModel.ready)
return;
if (!_grid)
onBackgroundReady()
if (type == InteractSignal.SCROLL || type == InteractSignal.ZOOM)
{
var oldHash:Point;
if (!_visibleBounds.containsRect(sceneModel.viewArea))
{
oldHash = _visibleHash.clone();
updateVisibleQuadrantArea();
updateVisibleQuadrants(oldHash);
}

if (type == InteractSignal.ZOOM)
_miniMapPresenter.updateScale();

if (Application.STATE == StateEvent.GAME_SECTOR)
{
if (!_quadrantBounds.contains(sceneModel.focus.x, sceneModel.focus.y))
{
var setView:SectorSetViewLocationRequest =
SectorSetViewLocationRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_SET_VIEW_LOCATION));
setView.x = sceneModel.focus.x;
setView.y = sceneModel.focus.y;
_serverController.send(setView);
updateQuadrantArea();
```

```
}
}
_miniMapPresenter.scrollMiniMapSignal.dispatch();
} else if (type == InteractSignal.RESOLUTION_CHANGE)
{
oldHash = _visibleHash.clone();
updateVisibleQuadrantArea();
updateVisibleQuadrants(oldHash);
}
}

/**
* Quadrant is the area we can observe; we inform the server of this so it will send us updates
within those bounds.
* This is 3 grid cells in each direction from our focal point.
*/
private function updateQuadrantArea():void
{
var focusHash:int = _grid.getHash(sceneModel.focus.x, sceneModel.focus.y);
_grid.getKey(focusHash, _point);

// Get the top left cell
_quadrantHash.x = _grid.getClampedCellByOffset(focusHash, -QUADRANT_RADIUS,
-QUADRANT_RADIUS);

// And the bottom right
_quadrantHash.y = _grid.getClampedCellByOffset(focusHash, +QUADRANT_RADIUS,
+QUADRANT_RADIUS);

var tlKey:Point = new Point();
var brKey:Point = new Point();
_grid.getKey(_quadrantHash.x, tlKey);
_grid.getKey(_quadrantHash.y, brKey);
brKey.x += _grid.CELL_SIZE;
brKey.y += _grid.CELL_SIZE;
_quadrantBounds.setTo(tlKey.x, tlKey.y, brKey.x - tlKey.x, brKey.y - tlKey.y);
}

/**
* Sets the visibility of the quadrants
*/
private function updateVisibleQuadrants( oldHash:Point ):void
{
var oldColumnSpan:Point = new Point(oldHash.x % _grid.COLUMNS, oldHash.y %
_grid.COLUMNS);
var newColumnSpan:Point = new Point(_visibleHash.x % _grid.COLUMNS, _visibleHash.y %
_grid.COLUMNS);
var oldRowSpan:Point = new Point((oldHash.x / _grid.COLUMNS) | 0, (oldHash.y /
_grid.COLUMNS) | 0);
var
```

```
newRowSpan:Point = new Point((_visibleHash.x / _grid.COLUMNS) | 0, (_visibleHash.y /
_grid.COLUMNS) | 0);

for (var i:int = oldRowSpan.x; i <= oldRowSpan.y; i++)
{
for (var j:int = oldColumnSpan.x; j <= oldColumnSpan.y; j++)
{
if (i < newRowSpan.x || i > newRowSpan.y || j < newColumnSpan.x || j > newColumnSpan.y)
{
adjustAnimations(_grid.getObjects(j + i * _grid.COLUMNS), false);
}
}
}

for (i = newRowSpan.x; i <= newRowSpan.y; i++)
{
for (j = newColumnSpan.x; j <= newColumnSpan.y; j++)
{
if (i < oldRowSpan.x || i > oldRowSpan.y || j < oldColumnSpan.x || j > oldColumnSpan.y)
{
adjustAnimations(_grid.getObjects(j + i * _grid.COLUMNS), true);
}
}
}
}

private function updateVisibleQuadrantArea():void
{
var viewArea:Rectangle = sceneModel.viewArea;
_visibleHash.x = _grid.getHash(viewArea.x, viewArea.y);
_visibleHash.y = _grid.getHash(viewArea.right, viewArea.bottom);
_grid.getKey(_visibleHash.x, _point);
_visibleBounds.setTo(_point.x, _point.y, 0, 0);
_grid.getKey(_visibleHash.y, _point);
_visibleBounds.setTo(_visibleBounds.x, _visibleBounds.y,
_point.x - _visibleBounds.x + _grid.CELL_SIZE,
_point.y - _visibleBounds.y + _grid.CELL_SIZE);
quadrantSignal.visibleHashChanged(_visibleBounds);
}

private function adjustAnimations( nodes:Array, visible:Boolean ):void
{
if (!nodes)
return;
for each (var node:IGridNode in nodes)
{
if (!visible)
{
//ensure that this object is actually not visible anymore
_testRect.setTo(node.position.x - node.animation.offsetX, node.position.y -
node.animation.offsetY,
```

```
node.animation.width, node.animation.height);
if (!_visibleBounds.intersects(_testRect))
node.animation.visible = false;
} else
node.animation.visible = true;
}
}

public function onBackgroundReady():void
{
_grid = new GridField(500, sceneModel.bounds);

updateQuadrantArea();
updateVisibleQuadrantArea();
_miniMapPresenter.mapWidth = quadrantSize;

for (var node:GridNode = nodes.head; node; node = node.next)
{
onEntityCreated(node);
}
quadrantSignal.visibleHashChanged(_visibleBounds);
}

/**
* The size of each axis of the quadrant. The quadrant is assumed square.
* Note that it can actually be smaller than this if we are close to the sector edge.
* We mainly use this for scaling the minimap.
*
* @return The coordinate size of the quadrant.
*/
public function get quadrantSize():int { return (QUADRANT_RADIUS * 2) * _grid.CELL_SIZE; }

[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set minimapPresenter( value:IMiniMapPresenter ):void { _miniMapPresenter =
value; }

override public function removeFromGame( game:Game ):void
{
_miniMapPresenter.clearMiniMapSignal.dispatch();

_game = null;
_newColumnSpan = null;
_newRowSpan = null;
_oldColumnSpan = null;
_oldRowSpan = null;
_point = null;
_quadrantHash = null;
_quadrantBounds = null;
_queue
```

```
= null;
_visibleHash = null;
_visibleBounds = null;

if (_grid)
_grid.destroy();
_grid = null;

interactSignal.remove(onScroll);
interactSignal = null;
quadrantSignal = null;
sceneModel = null;
nodes.nodeAdded.remove(onEntityCreated);
nodes.nodeRemoved.remove(onEntityRemoved);
nodes = null;
movingNodes = null;
}
}
}
```

File 278: igw\com\game\entity\systems\shared\render\Layers.as

```
package com.game.entity.systems.shared.render
{
import com.game.entity.nodes.shared.RenderNode;

public class Layers
{
private static const LAYERS:int = 16;

private var _allowDepth:Boolean;
private var _depth:int;
private var _depths:Array = [];
private var _gameLayer:*;
private var _i:int;
private var _layer:int;
private var _layers:Vector.<int>;
private var _length:int;
private var _render:*;

public function Layers( gameLayer:*, allowDepth:Boolean = true )
{
_allowDepth = allowDepth;
_gameLayer = gameLayer;
_layers = new Vector.<int>;
for (_i = 0; _i < LAYERS; _i++)
{
_depths[_i] = new Vector.<RenderNode>;
_layers[_i] = 0;
}
}
```

```
public function add( node:RenderNode ):void
{
_layer = node.position.layer;
if (_layer <= LAYERS)
{
//add to the correct depth
if (node.position.depth > -1)
{
_depth = node.position.depth;
_length = _depths[_layer].length;
for (_i = 0; _i < _length; _i++)
{
if (_depths[_layer][_i].position.depth > _depth || _depths[_layer][_i].position.depth == -1)
{
break;
}
}
_depths[_layer].splice(_i, 0, node);
if (_layer > 0)
_depth = _layers[_layer - 1] + _i;
else
_depth = _i;
} else
_depth = _layers[_layer];
//add to the layer
_render = node.animation.render;
if (_allowDepth)
{
if(_gameLayer.numChildren <= _depth)
_gameLayer.addChild(_render);
else
_gameLayer.addChildAt(_render, _depth);
}
else
_gameLayer.addChild(_render);
//update the next highest depths of subsequent layers
for (_i = _layer; _i < LAYERS; _i++)
{
_layers[_i]++;
}
}
}

public function remove( node:RenderNode ):void
{
_layer = node.position.layer;
if (_layer <= LAYERS)
{
//remove
```

```
the depth entry
if (node.position.depth > -1)
{
_length = _depths[_layer].length;
for (_i = 0; _i < _length; _i++)
{
if (_depths[_layer][_i] == node)
{
_depths[_layer].splice(_i, 1);
break;
}
}
}
_render = node.animation.render;
_gameLayer.removeChild(_render);
for (_i = _layer; _i < LAYERS; _i++)
{
_layers[_i]--;
if (_layers[_i] < 0)
throw new Error("wtf");
}
}
}

public function get scale():Number { return _gameLayer.scaleX }
public function set scale( v:Number ):void
{
if (_gameLayer)
{
_gameLayer.scaleX = v;
_gameLayer.scaleY = v;
}
}

public function destroy():void
{
_gameLayer = null;
}
}
}

-----------------------------------
File 279: igw\com\game\entity\systems\shared\render\RenderSystem.as
package com.game.entity.systems.shared.render
{
import com.Application;
import com.enum.CategoryEnum;
import com.enum.TypeEnum;
import com.event.signal.InteractSignal;
import com.game.entity.components.shared.Detail;
import
```

```
com.game.entity.components.shared.render.BarRender;
import com.game.entity.components.shared.render.BarRenderStarling;
import com.game.entity.components.shared.render.NameRender;
import com.game.entity.components.shared.render.NameRenderStarling;
import com.game.entity.components.shared.render.Render;
import com.game.entity.components.shared.render.RenderSprite;
import com.game.entity.components.shared.render.RenderSpriteStarling;
import com.game.entity.components.shared.render.RenderStarling;
import com.game.entity.nodes.shared.RenderNode;
import com.model.player.PlayerModel;
import com.model.scene.SceneModel;

import flash.geom.Rectangle;

import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;
import org.parade.core.IViewStack;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class RenderSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.shared.RenderNode")]
public var nodes:NodeList;
[Inject]
public var interactSignal:InteractSignal;
[Inject]
public var playerModel:PlayerModel;
[Inject]
public var sceneModel:SceneModel;
[Inject]
public var viewStack:IViewStack;

private var _backgroundLayer:Layers;
private var _dirty:Boolean;
private var _gameLayer:Layers;
//private var _away3D:Scene3D;
private var _zoomDirty:Boolean;

override public function addToGame( game:Game ):void
{
//_away3D = viewStack.getLayer(ViewEnum.AWAY3D_LAYER);
_backgroundLayer = new Layers(viewStack.getLayer(ViewEnum.BACKGROUND_LAYER));
_gameLayer = new Layers(viewStack.getLayer(ViewEnum.GAME_LAYER));
nodes.nodeRemoved.add(onEntityRemoved);

_dirty = _zoomDirty = false;
_gameLayer.scale = sceneModel.zoom;
interactSignal.add(onInteract);
}
```

```
override public function update( time:Number ):void
{
if (!sceneModel.ready)
return;

if (_zoomDirty)
{
_gameLayer.scale = sceneModel.zoom;
_zoomDirty = false;
}

var viewArea:Rectangle = sceneModel.viewArea;
for (var node:RenderNode = nodes.head; node; node = node.next)
{
if (!node.animation.visible)
{
if (node.render)
onEntityRemoved(node);
} else if (node.render || node.animation.visible)
{
if (!node.render)
renderNode(node);
else if (_dirty)
position(node, viewArea);
else if (node.position.dirty)
{
position(node, viewArea);
node.position.dirty = false;
}
if (node.position.depthDirty)
{
_gameLayer.remove(node);
_gameLayer.add(node);
node.position.depthDirty = false;
}
}
}
_dirty = false;
}

private function renderNode( node:RenderNode ):void
{
var rclass:Class = getRenderClass(node.detail);
node.render = ObjectPool.get(rclass);
/*if (node.detail.category == CategoryEnum.SHIP)
{
_gameLayer.add(node);
node.render3D = new Render3D();
_away3D.addChild(Render3D(node.render3D));
```

```
} else*/
if (node.detail.category == CategoryEnum.BACKGROUND)
_backgroundLayer.add(node);
else
_gameLayer.add(node);
position(node, sceneModel.viewArea);
node.render.alpha = node.animation.alpha;
if (node.animation.color != 0 && node.animation.color != 0xffffff && node.animation.color !=
node.render.color)
node.render.color = node.animation.color;
if (node.animation.blendMode != null)
node.render.blendMode = node.animation.blendMode;
node.render.updateFrame(node.animation.sprite, node.animation);
node.entity.add(node.render, rclass);
}

private function onEntityRemoved( node:RenderNode ):void
{
if (node.render)
{
/*if (node.detail.category == CategoryEnum.SHIP)
{
_gameLayer.remove(node);
//_away3D.removeChild(Render3D(node.render3D));
//node.render3D = null;
} else*/
if (node.detail.category == CategoryEnum.BACKGROUND)
_backgroundLayer.remove(node);
else
_gameLayer.remove(node);
ObjectPool.give(node.entity.remove(getRenderClass(node.detail)));
node.render = null;
}
}

private function position( node:RenderNode, viewArea:Rectangle ):void
{
node.render.x = node.position.x - node.animation.offsetX - viewArea.x *
node.position.parallaxSpeed;
node.render.y = node.position.y - node.animation.offsetY - viewArea.y *
node.position.parallaxSpeed;
/*if (node.render3D)
{
node.render3D.x = node.position.x - viewArea.x * node.position.parallaxSpeed;
node.render3D.y = node.position.y - viewArea.y * node.position.parallaxSpeed;
node.render3D.rotation = node.position.rotation * (180 / Math.PI);
}*/
if (node.animation.allowTransform)
node.render.applyTransform(node.position.rotation, node.animation.scaleX,
node.animation.scaleY,
```

```
node.animation.transformScaleFirst, node.animation.offsetX, node.animation.offsetY);
}

private function onInteract( type:String, dx:Number, dy:Number ):void
{
if (type == InteractSignal.SCROLL)
_dirty = true;
else if (type == InteractSignal.ZOOM)
{
_dirty = true;
_zoomDirty = true;
}
}

private function getRenderClass( detail:Detail ):Class
{
if (detail.category == CategoryEnum.BUILDING || detail.type == TypeEnum.FORCEFIELD ||
detail.category == CategoryEnum.DEBUFF)
return (Application.STARLING_ENABLED) ? RenderSpriteStarling : RenderSprite;
if (detail.type == TypeEnum.HEALTH_BAR || detail.type == TypeEnum.STATE_BAR)
return (Application.STARLING_ENABLED) ? BarRenderStarling : BarRender;
if (detail.type == TypeEnum.NAME)
return (Application.STARLING_ENABLED) ? NameRenderStarling : NameRender;
return (Application.STARLING_ENABLED) ? RenderStarling : Render;
}

public function get layers():Layers { return _gameLayer; }

override public function removeFromGame( game:Game ):void
{
nodes.nodeRemoved.remove(onEntityRemoved);
nodes = null;
sceneModel = null;
viewStack = null;
interactSignal.remove(onInteract);
interactSignal = null;
}
}
}

------------------------------------
File 280: igw\com\game\entity\systems\starbase\StarbaseSystem.as
package com.game.entity.systems.starbase
{
import com.Application;
import com.controller.sound.SoundController;
import com.enum.StarbaseConstructionEnum;
import com.enum.TypeEnum;
import com.event.StateEvent;
import com.game.entity.components.shared.Position;
import
```

```
com.game.entity.components.shared.Pylon;
import com.game.entity.components.shared.fsm.FSM;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.Platform;
import com.game.entity.factory.IInteractFactory;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.nodes.starbase.BuildingNode;
import com.game.entity.nodes.starbase.PlatformNode;
import com.model.battle.BattleModel;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.util.AllegianceUtil;
import com.util.statcalc.StatCalcUtil;

import flash.geom.Rectangle;
import flash.utils.Dictionary;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;
import org.console.Cc;
import org.shared.ObjectPool;
import org.starling.core.Starling;

public class StarbaseSystem extends System
{
public static const BUILDING_DAMAGED_HEALTH:Number = .6;
public static const DEPTH_SORT_ALL:String = "depthSortAll";
public static const DEPTH_SORT_BUILDINGS:String = "depthSortBuildings";
public static const DEPTH_SORT_PLATFORMS:String = "depthSortPlatforms";

[Inject(nodeType="com.game.entity.nodes.starbase.BuildingNode")]
public var buildingNodes:NodeList;

[Inject(nodeType="com.game.entity.nodes.starbase.PlatformNode")]
public var platformNodes:NodeList;

private var _battleModel:BattleModel;
private var _boundsRect1:Rectangle;
private var _boundsRect2:Rectangle;
private var _color:uint;
private var _depth:uint;
private var _depthDependency:Dictionary;
private var _depthVisited:Dictionary;
private var _forceFields:Dictionary;
private var _game:Game;
private var _generatorRanges:Vector.<Entity>;
private
```

```
var _generators:Vector.<BuildingNode>;
private var _interactFactory:IInteractFactory;
private var _pylonRanges:Vector.<Entity>;
private var _pylons:Vector.<BuildingNode>;
private var _resourceDepots:Vector.<BuildingNode>;
private var _soundController:SoundController;
private var _starbaseModel:StarbaseModel;
private var _starbaseFactory:IStarbaseFactory;
private var _turretRanges:Vector.<Entity>;
private var _turrets:Vector.<BuildingNode>;

override public function addToGame( game:Game ):void
{
buildingNodes.nodeAdded.add(onBuildingNodeAdded);
buildingNodes.nodeRemoved.add(onBuildingNodeRemoved);

_boundsRect1 = new Rectangle();
_boundsRect2 = new Rectangle();
_color = AllegianceUtil.instance.getFactionColor(CurrentUser.faction);
_depthDependency = new Dictionary();
_depthVisited = new Dictionary();
_forceFields = new Dictionary();
_generatorRanges = new Vector.<Entity>;
_generators = new Vector.<BuildingNode>;
_pylonRanges = new Vector.<Entity>;
_pylons = new Vector.<BuildingNode>;
_resourceDepots = new Vector.<BuildingNode>;
_turretRanges = new Vector.<Entity>;
_turrets = new Vector.<BuildingNode>;

_starbaseModel.addListener(updateResourceDepots);
Cc.addSlashCommand('forceContextLoss', forceContextLoss);
Cc.addSlashCommand('showGrid', onShowGrid);
}

public function createBuildingsFromStarbase():void
{
var buildings:Vector.<BuildingVO> = _starbaseModel.buildings;
for (var i:int = 0; i < buildings.length; i++)
{
if (buildings[i].constructionCategory == StarbaseConstructionEnum.PLATFORM)
_starbaseFactory.createBaseItem(buildings[i].id, buildings[i]);
else
_starbaseFactory.createBuilding(buildings[i].id, buildings[i]);
}
//show the platform
_starbaseFactory.createStarbasePlatform(CurrentUser.id);
findPylonConnections();
depthSort(DEPTH_SORT_ALL);
}

private
```

```
function onBuildingNodeAdded( node:BuildingNode ):void
{
node.init(onBuildingHealthChanged);
switch (node.building.buildingVO.itemClass)
{
case TypeEnum.POINT_DEFENSE_PLATFORM:
_turrets.push(node);
break;
case TypeEnum.PYLON:
node.$pylon.node = node;
_pylons.push(node);
break;
case TypeEnum.COMMAND_CENTER:
case TypeEnum.RESOURCE_DEPOT:
if (node.building.buildingVO.itemClass == TypeEnum.RESOURCE_DEPOT)
_resourceDepots.push(node);
else
_resourceDepots.unshift(node);
updateResourceDepots();
break;
case TypeEnum.SHIELD_GENERATOR:
_generators.push(node);
break;
}
}

private function onBuildingNodeRemoved( node:BuildingNode ):void
{
var index:int;
switch (node.building.buildingVO.itemClass)
{
case TypeEnum.FORCEFIELD:
if (node.entity.has(FSM))
ObjectPool.give(node.entity.remove(FSM));
break;
case TypeEnum.POINT_DEFENSE_PLATFORM:
index = _turrets.indexOf(node);
if (index > -1)
_turrets.splice(index, 1);
break;
case TypeEnum.PYLON:
index = _pylons.indexOf(node);
if (index > -1)
_pylons.splice(index, 1);
break;
case TypeEnum.COMMAND_CENTER:
case TypeEnum.RESOURCE_DEPOT:
index = _resourceDepots.indexOf(node);
if (index > -1)
{
_resourceDepots.splice(index,
```

```
1);
updateResourceDepots();
}
break;
case TypeEnum.SHIELD_GENERATOR:
index = _generators.indexOf(node);
if (index > -1)
_generators.splice(index, 1);
break;
}
node.destroy();
}

private function onBuildingHealthChanged( node:BuildingNode, percent:Number,
change:Number ):void
{
if (node.building.buildingVO.destroyed && percent > 0)
{
//restore the building from its' damaged state
_starbaseFactory.updateStarbaseBuilding(node.entity);
if (Application.STATE == StateEvent.GAME_STARBASE &&
node.building.buildingVO.itemClass == TypeEnum.PYLON)
findPylonConnections(node.entity, true);
} else if (node.building.buildingVO.damaged && percent >= BUILDING_DAMAGED_HEALTH)
{
//restore the building from its' damaged state
_starbaseFactory.updateStarbaseBuilding(node.entity);
} else if (percent < BUILDING_DAMAGED_HEALTH && !node.building.buildingVO.damaged)
{
_starbaseFactory.updateStarbaseBuilding(node.entity);
} else if (percent == 0 && !node.building.buildingVO.destroyed)
_starbaseFactory.updateStarbaseBuilding(node.entity);
}

private function updateResourceDepots():void
{
var cCredits:uint;
var cResources:uint;
var dCredits:uint;
var dResources:uint;
if (Application.STATE == StateEvent.GAME_BATTLE || Application.STATE ==
StateEvent.GAME_BATTLE_INIT)
{
dCredits = _battleModel.credits;
dResources = _battleModel.alloy + _battleModel.energy + _battleModel.synthetic;
} else
{
dCredits = _starbaseModel.currentBase.credits;
dResources = _starbaseModel.currentBase.alloy + _starbaseModel.currentBase.energy +
_starbaseModel.currentBase.synthetic;
}
```

```
var mCredits:uint;
var mResources:uint;
var percent:Number;
for (var i:int = 0; i < _resourceDepots.length; i++)
{
mCredits = StatCalcUtil.buildingStatCalc("CreditCap", _resourceDepots[i].building.buildingVO);
mResources = StatCalcUtil.buildingStatCalc("ResourceCap",
_resourceDepots[i].building.buildingVO) * 3;
cCredits = (dCredits < mCredits) ? dCredits : mCredits;
cResources = (dResources < mResources) ? dResources : mResources;
dCredits -= cCredits;
dResources -= cResources;
percent = (cCredits + cResources) / (mCredits + mResources);
//we only care about percents in 25% increments
if (percent >= 1)
percent = 1;
else if (percent >= .75)
percent = .75;
else if (percent >= .5)
percent = .5;
else if (percent >= .25)
percent = .25;
else
percent = 0;
if (_resourceDepots[i].building.buildingVO.percentFilled != percent)
{
_resourceDepots[i].building.buildingVO.percentFilled = percent;
if (i > 0)
_starbaseFactory.updateStarbaseBuilding(_resourceDepots[i].entity);
}
}
}


//=============================================================================
//*****************************************************************************************************
// Point Defense Platforms
//*****************************************************************************************************

//=============================================================================

public function showTurretRanges( target:Entity, show:Boolean = true ):void
{
if (_turretRanges.length > 0 && !show)
{
//remove the ranges if needed
for (var i:int = 0; i < _turretRanges.length; i++)
_interactFactory.destroyInteractEntity(_turretRanges[i]);
_turretRanges.length = 0;
}
```

```
//cycle through the existing turrets and add ranges to those that need
if (show)
{
if (target == null)
{
for (i = 0; i < _turrets.length; i++)
_turretRanges.push(_interactFactory.createRange(_turrets[i].entity));
} else
_turretRanges.push(_interactFactory.createRange(target));
}
}


//=============================================================================
//*****************************************************************************
// Pylon
//*****************************************************************************

//=============================================================================

public function showPylonRanges( target:Entity, show:Boolean = true ):void
{
if (_pylonRanges.length > 0 && !show)
{
//remove the ranges if needed
for (var i:int = 0; i < _pylonRanges.length; i++)
_interactFactory.destroyInteractEntity(_pylonRanges[i]);
_pylonRanges.length = 0;
}

//cycle through the existing pylons and add ranges to those that need
if (show)
{
if (target == null)
{
for (i = 0; i < _pylons.length; i++)
_pylonRanges.push(_interactFactory.createRange(_pylons[i].entity));
} else
_pylonRanges.push(_interactFactory.createRange(target));
}
}

public function positionPylonBase( entity:Entity ):void
{
var building:Building = entity.get(Building);
var position:Position = entity.get(Position);
var pylon:Pylon = entity.get(Pylon);

pylon.baseX
```

```
= building.buildingVO.baseX;
pylon.baseY = building.buildingVO.baseY;
var platform:Platform = pylon.bottom.get(Platform);
platform.buildingVO.baseX = building.buildingVO.baseX;
platform.buildingVO.baseY = building.buildingVO.baseY;
var positionB:Position = pylon.bottom.get(Position);
positionB.x = position.x;
positionB.y = position.y;
depthSort(DEPTH_SORT_PLATFORMS);
}

public function findPylonConnections( target:Entity = null, disable:Boolean = false ):void
{
if (target)
findPylonConnectionsForEntity(target, disable);
else
findPylonConnectionsForAll();
}

private function findPylonConnectionsForAll():void
{
var pylonA:BuildingNode;
var pylonB:BuildingNode;

for (var i:int = 0; i < _pylons.length; i++)
{
pylonA = _pylons[i];
if (pylonA.building.buildingVO.currentHealth < 1 || !pylonA.building.buildingVO.built)
continue;
for (var j:int = i + 1; j < _pylons.length; j++)
{
pylonB = _pylons[j];
if (pylonB.building.buildingVO.currentHealth < 1 || !pylonB.building.buildingVO.built)
continue;
checkConnection(pylonA, pylonB);
}
}
}

private function findPylonConnectionsForEntity( target:Entity, disable:Boolean ):void
{
var pylonA:BuildingNode = Pylon(target.get(Pylon)).node;
var pylonB:BuildingNode;

if (disable)
{
if (pylonA.$pylon.bottomConnection)
{
_starbaseFactory.destroyStarbaseItem(_game.getEntity(pylonA.$pylon.bottomWallKey));
pylonA.$pylon.removeConnection(pylonA.$pylon.bottomConnection);
}
```

```
if (pylonA.$pylon.leftConnection)
{
_starbaseFactory.destroyStarbaseItem(_game.getEntity(pylonA.$pylon.leftWallKey));
pylonA.$pylon.removeConnection(pylonA.$pylon.leftConnection);
}
if (pylonA.$pylon.rightConnection)
{
_starbaseFactory.destroyStarbaseItem(_game.getEntity(pylonA.$pylon.rightWallKey));
pylonA.$pylon.removeConnection(pylonA.$pylon.rightConnection);
}
if (pylonA.$pylon.topConnection)
{
_starbaseFactory.destroyStarbaseItem(_game.getEntity(pylonA.$pylon.topWallKey));
pylonA.$pylon.removeConnection(pylonA.$pylon.topConnection);
}
return;
}

if (pylonA.building.buildingVO.currentHealth < 1 || !pylonA.building.buildingVO.built)
return;

//update pylon position to match building
pylonA.$pylon.baseX = pylonA.building.buildingVO.baseX;
pylonA.$pylon.baseY = pylonA.building.buildingVO.baseY;

for (var i:int = 0; i < _pylons.length; i++)
{
pylonB = _pylons[i];
if (pylonB == pylonA || pylonB.building.buildingVO.currentHealth < 1 ||
!pylonB.building.buildingVO.built)
continue;
checkConnection(pylonA, pylonB, true);
}
}

private function checkConnection( pylonA:BuildingNode, pylonB:BuildingNode,
checkBOnNoConnection:Boolean = false ):void
{
var distance:int;
var j:int;
var result:Object;
var maxLength:int;
var startX:int;
var startY:int;
var endX:int;
var endY:int;

//ensure at least one of the pylons is in range of the other
distance = -1;
if
```

```
(pylonA.building.buildingVO.baseX == pylonB.building.buildingVO.baseX)
distance = Math.abs(pylonA.building.buildingVO.baseY - pylonB.building.buildingVO.baseY);
else if (pylonA.building.buildingVO.baseY == pylonB.building.buildingVO.baseY)
distance = Math.abs(pylonA.building.buildingVO.baseX - pylonB.building.buildingVO.baseX);
distance *= 18;
maxLength = Math.max(pylonA.building.buildingVO.shieldRadius * 90,
pylonB.building.buildingVO.shieldRadius * 90);
if (distance > 90 && distance <= maxLength)
{
//one final check to ensure there is not a building in the path of the forcefield
startX = pylonA.$pylon.baseX < pylonB.$pylon.baseX ? pylonA.$pylon.baseX :
pylonB.$pylon.baseX;
if (pylonA.$pylon.baseX != pylonB.$pylon.baseX)
startX += 5;
startY = pylonA.$pylon.baseY < pylonB.$pylon.baseY ? pylonA.$pylon.baseY :
pylonB.$pylon.baseY;
if (pylonA.$pylon.baseY != pylonB.$pylon.baseY)
startY += 5;
endX = Math.abs(pylonA.$pylon.baseX - pylonB.$pylon.baseX) - 10;
endX = (endX < 0) ? 5 : endX + 5;
endY = Math.abs(pylonA.$pylon.baseY - pylonB.$pylon.baseY) - 10;
endY = (endY < 0) ? 5 : endY + 5;
_boundsRect1.setTo(startX, startY, endX, endY);
for (var node:BuildingNode = buildingNodes.head; node; node = node.next)
{
if (node.building.buildingVO.itemClass == TypeEnum.FORCEFIELD || node == pylonA || node
== pylonB)
continue;
_boundsRect2.setTo(node.building.buildingVO.baseX, node.building.buildingVO.baseY,
node.building.buildingVO.sizeX, node.building.buildingVO.sizeY);
if (_boundsRect1.intersects(_boundsRect2) || _boundsRect1.containsRect(_boundsRect2))
{
//remove the connecting forcefield because there is a building in the way
pylonA.$pylon.removeConnection(pylonB);
if (_game.getEntity(pylonA.$pylon.craftKey(pylonB)))
_starbaseFactory.destroyStarbaseItem(_game.getEntity(pylonA.$pylon.craftKey(pylonB)));
return;
}
}

//there is no building in the way so add the connection
result = pylonA.$pylon.addConnection(pylonB);
if (result)
{
if (result.added)
_starbaseFactory.createForcefield(result.added, pylonA.$pylon, pylonB.$pylon, _color);
if (result.removed)
_starbaseFactory.destroyStarbaseItem(_game.getEntity(result.removed));
}
result = pylonB.$pylon.addConnection(pylonA);
if
```

```
(result)
{
if (result.added)
_starbaseFactory.createForcefield(result.added, pylonA.$pylon, pylonB.$pylon, _color);
if (result.removed)
_starbaseFactory.destroyStarbaseItem(_game.getEntity(result.removed));
}
} else if (_game.getEntity(pylonA.$pylon.craftKey(pylonB)))
{
pylonA.$pylon.removeConnection(pylonB);
_starbaseFactory.destroyStarbaseItem(_game.getEntity(pylonA.$pylon.craftKey(pylonB)));
if (checkBOnNoConnection)
findPylonConnections(pylonB.entity);
}
}


//=============================================================================
//*****************************************************************************************************
// Shield Generators
//*****************************************************************************************************

//=============================================================================

public function showShieldRanges( target:Entity, show:Boolean = true ):void
{
if (_generatorRanges.length > 0 && !show)
{
//remove the ranges if needed
for (var i:int = 0; i < _generatorRanges.length; i++)
_interactFactory.destroyInteractEntity(_generatorRanges[i]);
_generatorRanges.length = 0;
}

//cycle through the existing generators and add ranges to those that need
if (show)
{
if (target == null)
{
for (i = 0; i < _generators.length; i++)
_generatorRanges.push(_interactFactory.createRange(_generators[i].entity));
} else
_generatorRanges.push(_interactFactory.createRange(target));
}
}

public function showShields( forceOff:Boolean = false, selectedGenerator:Entity = null ):void
{
//brute force checks to see if a building has a shield
var active:Boolean;
var
```

```
distance:Number;
var distanceX:Number;
var distanceY:Number;
var generator:BuildingNode;
var module:IPrototype;
var position:Position;
var size:Number;
var slot:String;
var type:String;
for (var node:BuildingNode = buildingNodes.head; node; node = node.next)
{
active = false;
size = node.building.buildingVO.sizeX * .5;
type = node.building.buildingVO.itemClass == TypeEnum.POINT_DEFENSE_PLATFORM ?
TypeEnum.TURRET_SHIELD : TypeEnum.BUILDING_SHIELD;
if (!forceOff && node.building.buildingVO.itemClass != TypeEnum.FORCEFIELD)
{
for (var i:int = 0; i < _generators.length; i++)
{
if (_generators[i] != node && (selectedGenerator == null || selectedGenerator ==
_generators[i].entity))
{
slot = _generators[i].building.buildingVO.getValue("slots")[0];
module = (_generators[i].building.buildingVO.modules.hasOwnProperty(slot)) ?
_generators[i].building.buildingVO.modules[slot] : null;
if (module)
{
generator = _generators[i];
distanceX = (node.building.buildingVO.baseX + size) - (generator.building.buildingVO.baseX +
2.5);
distanceY = (node.building.buildingVO.baseY + size) - (generator.building.buildingVO.baseY +
2.5);
distance = Math.sqrt((distanceX * distanceX) + (distanceY * distanceY));
if (distance < (Math.round(generator.building.buildingVO.shieldRadius / 18)))
{
active = true;
break;
}
}
}
}
}
if (active)
{
if (node.$vcList && !node.$vcList.hasComponentType(type))
node.$vcList.addComponentType(type);
} else if (node.$vcList)
node.$vcList.removeComponentType(type);
}
}


//=========================================================================
```

```
//**************************************************************************************************************
// Depth Sorting
//**************************************************************************************************************

//==============================================================================

public function depthSort( type:String ):void
{
if (type == DEPTH_SORT_ALL || type == DEPTH_SORT_PLATFORMS)
depthSortPlatforms();
if (type == DEPTH_SORT_ALL || type == DEPTH_SORT_BUILDINGS)
depthSortBuildings();
}

private function depthSortPlatforms():void
{
var platformNode:PlatformNode;
var platformNodeInner:PlatformNode;
var vo:BuildingVO;
var voInner:BuildingVO;
for (platformNode = platformNodes.head; platformNode; platformNode = platformNode.next)
{
var behind:Array = [];
vo = platformNode.platform.buildingVO;
var rightA:Number = vo.baseX + vo.sizeX;
var frontA:Number = vo.baseY + vo.sizeY;

for (platformNodeInner = platformNodes.head; platformNodeInner; platformNodeInner =
platformNodeInner.next)
{
voInner = platformNodeInner.platform.buildingVO;

// See if B should go behind A
// simplest possible check, interpenetrations also count as "behind", which does do a bit more
work later, but the inner loop tradeoff for a faster check makes up for it
if ((voInner.baseX < rightA) &&
(voInner.baseY < frontA) &&
(platformNode !== platformNodeInner))
{
behind.push(platformNodeInner);
}
}

_depthDependency[platformNode] = behind;
}
// Set the childrens' depth, using dependency ordering
_depth = 0;
for (platformNode = platformNodes.head; platformNode; platformNode = platformNode.next)
if (true !== _depthVisited[platformNode])
place(platformNode);
```

```
// Clear out temporary dictionary so we're not retaining memory between calls
_depthDependency = new Dictionary();
_depthVisited = new Dictionary();
}

private function depthSortBuildings():void
{
var buildingNode:BuildingNode;
var buildingNodeInner:BuildingNode;
var vo:BuildingVO;
var voInner:BuildingVO;
for (buildingNode = buildingNodes.head; buildingNode; buildingNode = buildingNode.next)
{
var behind:Array = [];
vo = buildingNode.building.buildingVO;
var rightA:Number = vo.baseX + vo.sizeX;
var frontA:Number = vo.baseY + vo.sizeY;

for (buildingNodeInner = buildingNodes.head; buildingNodeInner; buildingNodeInner =
buildingNodeInner.next)
{
voInner = buildingNodeInner.building.buildingVO;

// See if B should go behind A
// simplest possible check, interpenetrations also count as "behind", which does do a bit more
work later, but the inner loop tradeoff for a faster check makes up for it
if ((voInner.baseX < rightA) &&
(voInner.baseY < frontA) &&
(buildingNode != buildingNodeInner))
{
behind.push(buildingNodeInner);
}
}

_depthDependency[buildingNode] = behind;
}
// Set the childrens' depth, using dependency ordering
_depth = 0;
for (buildingNode = buildingNodes.head; buildingNode; buildingNode = buildingNode.next)
if (true !== _depthVisited[buildingNode])
place(buildingNode);

// Clear out temporary dictionary so we're not retaining memory between calls
_depthDependency = new Dictionary();
_depthVisited = new Dictionary();
}

/**
*
```

```
Dependency-ordered depth placement of the given objects and its dependencies.
*/
private function place( nodeToPlace:* ):void
{
_depthVisited[nodeToPlace] = true;

var node:*;
for (var i:int = 0; i < _depthDependency[nodeToPlace].length; i++)
{
node = _depthDependency[nodeToPlace][i];
if (true !== _depthVisited[node])
place(node);
}

nodeToPlace.position.depth = _depth;

_depth += 1;
}

private function onShowGrid():void
{
_starbaseModel.grid.showGrid(_game, _starbaseFactory);
}

private function forceContextLoss():void
{
Starling.current.context.dispose();
}

[Inject]
public function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
public function get generators():Vector.<BuildingNode> { return _generators; }
[Inject]
public function set interactFactory( v:IInteractFactory ):void { _interactFactory = v; }
[Inject]
public function set soundController( v:SoundController ):void { _soundController = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }

override public function removeFromGame( game:Game ):void
{
showShieldRanges(null, false);
showTurretRanges(null, false);
buildingNodes.nodeAdded.remove(onBuildingNodeAdded);
buildingNodes.nodeRemoved.remove(onBuildingNodeRemoved);
buildingNodes = null;
_boundsRect1
```

```
= null;
_boundsRect2 = null;
_depthDependency = null;
_generatorRanges = null;
_generators.length = 0;
_generators = null;
_pylonRanges = null;
_pylons.length = 0;
_pylons = null;
_resourceDepots.length = 0;
_resourceDepots = null;
_soundController = null;
_turretRanges = null;
_turrets.length = 0;
_turrets = null;
platformNodes = null;
_depthVisited = null;
_game = null;
_starbaseModel.removeListener(updateResourceDepots);
}
}
}


---------------------------------
File 281: igw\com\game\entity\systems\starbase\StateSystem.as
package com.game.entity.systems.starbase
{
import com.Application;
import com.controller.transaction.TransactionController;
import com.enum.TypeEnum;
import com.event.TransactionEvent;
import com.event.signal.TransactionSignal;
import com.game.entity.components.shared.Animation;
import com.game.entity.components.shared.VCList;
import com.game.entity.components.shared.fsm.FSM;
import com.game.entity.components.starbase.Construction;
import com.game.entity.components.starbase.State;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.nodes.starbase.StateNode;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;

import flash.events.IEventDispatcher;
import flash.utils.Dictionary;

import org.adobe.utils.StringUtil;
import
```

```
org.ash.core.Entity;
import org.ash.core.Game;
import org.ash.core.NodeList;
import org.ash.core.System;
import org.shared.ObjectPool;

/**
* Shows progress bars over buildings and handles the animation sequence
* for building construction and repair
*/
public class StateSystem extends System
{
[Inject(nodeType="com.game.entity.nodes.starbase.StateNode")]
public var nodes:NodeList;

private var _eventDispatcher:IEventDispatcher;
private var _game:Game;
private var _starbaseFactory:IStarbaseFactory;
private var _starbaseModel:StarbaseModel;
private var _time:Number;
private var _transactionController:TransactionController;
private var _transactionModel:TransactionModel;

override public function addToGame( game:Game ):void
{
_game = game;
_time = 0;

//add the transaction listeners
_transactionModel.addListener(TransactionSignal.TRANSACTION_REMOVED,
onTransactionRemoved);
_transactionModel.addListener(TransactionSignal.TRANSACTION_UPDATED,
onTransactionUpdated);

//go through the current transactions and set states
var transactions:Dictionary = _transactionModel.transactions;
for each (var transaction:TransactionVO in transactions)
{
onTransactionUpdated(transaction);
}
}

/**
* Acts as a state machine for building construction animation
* and also updates the progress bars of transactions
*/
override public function update( time:Number ):void
{
_time += time;
//update the progress bars if there are any
if
```

```
(nodes.head && _time >= 1)
{
var animation:Animation;
var state:State;
for (var node:StateNode = nodes.head; node; node = node.next)
{
state = node.state;
state.text = StringUtil.getBuildTime(state.remainingTime * .001, (Application.LANGUAGE ==
'en') ? 2 : 1);
if (state.component)
{
animation = state.component.get(Animation);
animation.scaleX = state.percentageDone;
animation.text = state.text;
if (animation.render)
animation.render.scaleX = state.percentageDone;
}
}
_time -= 1;
}
}

private function onTransactionUpdated( transaction:TransactionVO ):void
{
//ignore transactions that were instantly built
if (transaction.timeMS <= 0)
return;
var animation:Animation;
var entity:Entity = getEntity(transaction);
if (entity)
{
var vcList:VCList = entity.get(VCList);
var state:State = entity.get(State);
if (state == null)
{
state = ObjectPool.get(State);
entity.add(state, State);
}

state.addTransaction(transaction);
updateState(entity);
}
}

private function onTransactionRemoved( transaction:TransactionVO ):void
{
//ignore transactions that were instantly built
if (transaction.timeMS == 0)
return;
var entity:Entity = getEntity(transaction);
if
```

```
(entity && entity.has(State))
{
var state:State = entity.get(State);
state.removeTransaction(transaction.id);
updateState(entity);
}
}

private function updateState( entity:Entity ):void
{
var animation:Animation;
var state:State = entity.get(State);
var vcList:VCList = entity.get(VCList);

if (!state.showConstruction)
{
var construction:Entity = vcList.getComponent(TypeEnum.BUILDING_CONSTRUCTION);
//set the state of the building construction if it has one
if (construction)
FSM(construction.get(FSM)).state = Construction.BEAM_ASCEND;
else //call the remove, just in case the transaction finished before we had time to create the
construction animation
vcList.removeComponentType(TypeEnum.BUILDING_CONSTRUCTION);
} else if (state.showConstruction)
vcList.addComponentType(TypeEnum.BUILDING_CONSTRUCTION);

if (state.transactionCount <= 0)
{
vcList.removeComponentType(TypeEnum.STATE_BAR);
ObjectPool.give(entity.remove(State));
_starbaseFactory.updateStarbaseBuilding(entity);
} else
{
vcList.addComponentType(TypeEnum.STATE_BAR);
state.text = StringUtil.getBuildTime(state.remainingTime * .001, (Application.LANGUAGE ==
'en') ? 2 : 1);
if (state.component)
{
animation = state.component.get(Animation);
animation.scaleX = state.percentageDone;
animation.text = state.text;
if (animation.render)
animation.render.scaleX = state.percentageDone;
}
}
}

private function getEntity( transaction:TransactionVO ):Entity
{
var buildingVO:BuildingVO;
var
```

```
entity:Entity = _game.getEntity(transaction.id);
if (!entity)
{
//if we haven't found an entity this may be a research or something related to a ship or fleet.
//find the associated building.
if (transaction.baseID == _starbaseModel.currentBase.id)
{
switch (transaction.type)
{
case TransactionEvent.STARBASE_RESEARCH:
var researchVO:ResearchVO = _starbaseModel.getResearchByID(transaction.id);
if (researchVO)
{
entity =
_game.getEntity(_starbaseModel.getBuildingByClass(researchVO.requiredBuildingClass).id);
}
break;
case TransactionEvent.STARBASE_REFIT_SHIP:
case TransactionEvent.STARBASE_RECYCLE_SHIP:
case TransactionEvent.STARBASE_BUILD_SHIP:
buildingVO = _starbaseModel.getBuildingByClass(TypeEnum.CONSTRUCTION_BAY);
if (buildingVO)
entity = _game.getEntity(buildingVO.id);
break;
case TransactionEvent.STARBASE_REPAIR_FLEET:
buildingVO = _starbaseModel.getBuildingByClass(TypeEnum.DOCK);
if (buildingVO)
entity = _game.getEntity(buildingVO.id);
break;
}

}
}
if (entity && entity.has(VCList))
return entity;
return null;
}

[Inject]
public function set eventDispatcher( v:IEventDispatcher ):void { _eventDispatcher = v; }
[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set transactionModel( v:TransactionModel ):void { _transactionModel = v; }

override
```

```
public function removeFromGame( game:Game ):void
{
nodes = null;
_eventDispatcher = null;

//remove the transaction listeners
_transactionModel.removeListener(onTransactionRemoved);
_transactionModel.removeListener(onTransactionUpdated);

_game = null;
_starbaseModel = null;
_transactionController = null;
_transactionModel = null;
}
}
}
```

File 282: igw\com\model\Model.as

```
package com.model
{
import flash.events.Event;
import flash.events.IEventDispatcher;

public class Model
{
protected var _eventDispatcher:IEventDispatcher;

/**
* Dispatch helper method
*
* @param event The Event to dispatch on the <code>IContext</code>'s
<code>IEventDispatcher</code>
*/
protected function dispatch( event:Event ):Boolean
{
if (_eventDispatcher.hasEventListener(event.type))
return _eventDispatcher.dispatchEvent(event);
return false;
}

[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
}
}
```

File 283: igw\com\model\achievements\AchievementModel.as

```
package
```

```
com.model.achievements
{
import com.enum.ToastEnum;
import com.event.ToastEvent;
import com.model.Model;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.AchievementData;
import com.service.server.incoming.data.ScoreData;
import com.service.server.incoming.data.MissionScoreData;
import com.service.server.incoming.starbase.StarbaseAchievementsResponse;
import com.service.server.incoming.starbase.StarbaseAllScoresResponse;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;

public class AchievementModel extends Model
{
public var onAchievementsUpdated:Signal;
public var onAllScoresUpdated:Signal;

private var _achievements:Dictionary;
private var _scores:Dictionary;
private var _missionScores:Dictionary;


public function AchievementModel()
{
super();

_achievements = new Dictionary();
_scores = new Dictionary();
_missionScores = new Dictionary();
onAchievementsUpdated = new Signal(Dictionary);
onAllScoresUpdated = new Signal(Dictionary);
}

public function addData( achievementResponse:StarbaseAchievementsResponse ):void
{
var i:uint;
var currentAchievementData:AchievementData;
var currentScoreData:ScoreData;
var currentAchievement:AchievementVO;
var currentScore:ScoreVO;
var achievements:Vector.<AchievementData> = achievementResponse.achievements;
var scores:Vector.<ScoreData> = achievementResponse.scores;
var len:uint = scores.length;

for (; i < len; ++i)
{
currentScoreData
```

```
= scores[i];
currentScore = new ScoreVO(currentScoreData.key, currentScoreData.scoreKey,
currentScoreData.value);
_scores[currentScoreData.scoreKey] = currentScore;
}

len = achievements.length;
for (i = 0; i < len; ++i)
{
currentAchievementData = achievements[i];
currentAchievement = new AchievementVO(currentAchievementData.key,
currentAchievementData.achievementPrototype, currentAchievementData.claimedFlag);
_achievements[currentAchievementData.key] = currentAchievement;

if (achievementResponse.unlockToast)
popToast(currentAchievement);
}


onAchievementsUpdated.dispatch(_achievements, _scores);
}

public function addAllScoreData( allScoresResponse:StarbaseAllScoresResponse ):void
{
var i:uint;
var currentScoreData:ScoreData;
var currentMissionScoreData:MissionScoreData;
var currentScore:ScoreVO;
var currentMissionScore:MissionScoreVO;
var scores:Vector.<ScoreData> = allScoresResponse.scores;
var missionScores:Vector.<MissionScoreData> = allScoresResponse.missionScores;

var len:uint = scores.length;
for (; i < len; ++i)
{
currentScoreData = scores[i];
currentScore = new ScoreVO(currentScoreData.key, currentScoreData.scoreKey,
currentScoreData.value);
_scores[currentScoreData.scoreKey] = currentScore;
}

len = missionScores.length;
for (; i < len; ++i)
{
currentMissionScoreData = missionScores[i];
currentMissionScore = new MissionScoreVO(currentMissionScoreData.key,
currentMissionScoreData.instancedMissionID, currentMissionScoreData.bestTime);
_missionScores[currentMissionScoreData.instancedMissionID] = currentMissionScore;
}

onAllScoresUpdated.dispatch(_scores,
```

```
_missionScores);
}

public function getScoreValueByName( v:String ):uint
{
if (v in _scores)
return _scores[v].value;

return 0;
}

private function popToast( achievement:AchievementVO ):void
{
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = ToastEnum.ACHIEVEMENT;
toastEvent.prototype =
PrototypeModel.instance.getAchievementPrototypeByName(achievement.achievementPrototype);
dispatch(toastEvent);
}

}
}

-------------------------------------
File 284: igw\com\model\achievements\AchievementVO.as
package com.model.achievements
{
public class AchievementVO
{
private var _key:String;
private var _category:String;
private var _achievementPrototype:String;
private var _claimedFlag:Boolean;

public function AchievementVO( key:String, achievementPrototype:String, claimedFlag:Boolean
)
{
_key = key;
_category = achievementPrototype.slice(0, achievementPrototype.length - 2);
_achievementPrototype = achievementPrototype;
_claimedFlag = claimedFlag;
}

public function get key():String { return _key; }
public function get category():String { return _category; }
public function get achievementPrototype():String { return _achievementPrototype; }
public function set claimedFlag( v:Boolean ):void { _claimedFlag = v; }
public function get claimedFlag():Boolean { return _claimedFlag; }
}
}

-------------------------------------
```

File 285: igw\com\model\achievements\MissionScoreVO.as

```
package com.model.achievements
{
public class MissionScoreVO
{
private var _key:String;
private var _instancedMissionID:int;
private var _bestTime:int;

public function MissionScoreVO( key:String, instancedMissionID:int, bestTime:int )
{
_key = key;
_instancedMissionID = instancedMissionID;
_bestTime = bestTime;
}

public function get key():String { return _key; }
public function get instancedMissionID():int { return _instancedMissionID; }
public function get bestTime():int { return _bestTime; }
}
}
```

File 286: igw\com\model\achievements\ScoreVO.as

```
package com.model.achievements
{
public class ScoreVO
{
private var _key:String;
private var _scoreKey:String;
private var _value:Number;

public function ScoreVO( key:String, scoreKey:String, value:Number )
{
_key = key;
_scoreKey = scoreKey;
_value = value;
}

public function get key():String { return _key; }
public function get scoreKey():String { return _scoreKey; }
public function get value():Number { return _value; }
}
}
```

File 287: igw\com\model\alliance\AllianceInviteVO.as

```
package com.model.alliance
{
public class AllianceInviteVO
{
```

```
private var _inviterKey:String;
private var _inviterName:String;
private var _alliance:AllianceVO;

public function AllianceInviteVO( inviterKey:String, inviterName:String, alliance:AllianceVO )
{
_inviterKey = inviterKey;
_inviterName = inviterName;
_alliance = alliance;
}

public function get inviterKey():String { return _inviterKey; }
public function get inviterName():String { return _inviterName; }
public function get alliance():AllianceVO { return _alliance; }
public function set alliance( v:AllianceVO ):void { _alliance = v; }
public function get allianceMembers():Vector.<AllianceMemberVO> { return _alliance.members;
}
public function set allianceMembers( v:Vector.<AllianceMemberVO> ):void { _alliance.members
= v; }
public function get allianceKey():String { return (_alliance != null) ? _alliance.key : ''; }
}
}

-----------------------------------
File 288: igw\com\model\alliance\AllianceMemberVO.as
package com.model.alliance
{
public class AllianceMemberVO
{
private var _key:String;
private var _name:String;
private var _xp:uint;
private var _lastOnline:uint;
private var _rank:int;

public function AllianceMemberVO( key:String, name:String, xp:uint, lastOnline:uint, rank:int )
{
_key = key;
_name = name;
_xp = xp;
_lastOnline = lastOnline;
_rank = rank;
}

public function get key():String { return _key; }
public function get name():String { return _name; }
public function get xp():uint { return _xp; }
public function get lastOnline():uint { return _lastOnline; }
public function get rank():int { return _rank; }
}
```

```
}

-----------------------------------
File 289: igw\com\model\alliance\AllianceModel.as
package com.model.alliance
{
import com.enum.server.AllianceResponseEnum;
import com.model.Model;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;

public class AllianceModel extends Model
{
private var _alliances:Dictionary;
private var _openAlliances:Dictionary;
private var _invitedAlliances:Dictionary;
private var _emailInvites:Dictionary;

public var onOpenAlliancesUpdated:Signal;
public var onInvitedAlliancesUpdated:Signal;
public var onAllianceUpdated:Signal;
public var onAllianceMembersUpdated:Signal;
public var onGenericAllianceMessageRecieved:Signal;

public function AllianceModel()
{
super();

_alliances = new Dictionary();
_openAlliances = new Dictionary();
_invitedAlliances = new Dictionary();
onOpenAlliancesUpdated = new Signal(Dictionary);
onInvitedAlliancesUpdated = new Signal(Dictionary);
onAllianceUpdated = new Signal(String, AllianceVO);
onAllianceMembersUpdated = new Signal(String, Vector.<AllianceMemberVO>);
onGenericAllianceMessageRecieved = new Signal(int, String);
}

public function addAlliance( alliance:AllianceVO ):void
{
if (alliance)
{
var key:String = alliance.key;
if (key != '')
{
if (key in _alliances)
{
var
```

```
oldAllianceData:AllianceVO = _alliances[key];
alliance.members = oldAllianceData.members;
}
_alliances[key] = alliance;
onAllianceUpdated.dispatch(key, alliance);
}
}
}

public function addOpenAlliances( alliances:Vector.<AllianceVO> ):void
{
if (alliances && alliances.length > 0)
{
var len:uint = alliances.length;
var alliance:AllianceVO;
var key:String;
var oldAllianceData:AllianceVO;
for (var i:uint = 0; i < len; ++i)
{
alliance = alliances[i];
key = alliance.key;
if (key != '')
{
if (key in alliances)
{
oldAllianceData = alliances[key];
alliance.members = oldAllianceData.members;
}
_openAlliances[key] = alliance;
onAllianceUpdated.dispatch(key, alliance);
}
}
onOpenAlliancesUpdated.dispatch(_openAlliances);
}
}

public function addInvitedAlliance( alliance:AllianceInviteVO ):void
{
if (alliance)
{
var oldAllianceData:AllianceInviteVO;
var key:String = alliance.allianceKey;
if (key != '')
{
if (key in _invitedAlliances)
{
oldAllianceData = _invitedAlliances[key];
alliance.allianceMembers = oldAllianceData.allianceMembers;
}
_invitedAlliances[key] = alliance;
}
```

```
onInvitedAlliancesUpdated.dispatch(_invitedAlliances);
onGenericAllianceMessageRecieved.dispatch(AllianceResponseEnum.INVITED, key);
}
}


public function updateMembers( key:String, members:Vector.<AllianceMemberVO> ):void
{
if (key != '')
{
if (key in _alliances)
{
var allianceData:AllianceVO = _alliances[key];
allianceData.members = members;
_alliances[key] = allianceData;
onAllianceUpdated.dispatch(key, allianceData);
}

if (key in _openAlliances)
{
var openAllianceData:AllianceVO = _openAlliances[key];
openAllianceData.members = members;
_openAlliances[key] = openAllianceData;
onAllianceUpdated.dispatch(key, openAllianceData);
}

if (key in _invitedAlliances)
{
var inviteAllianceData:AllianceInviteVO = _invitedAlliances[key];
inviteAllianceData.allianceMembers = members;
_invitedAlliances[key] = inviteAllianceData;

}
onAllianceMembersUpdated.dispatch(key, members);
}
}

public function handleGenericMessage( messageEnum:int, allianceKey:String ):void
{
onGenericAllianceMessageRecieved.dispatch(messageEnum, allianceKey);
}

public function getAllianceInvites():Dictionary { return _invitedAlliances; }

public function setEmailInvites ( emailInvites:Dictionary ):void
{
_emailInvites = emailInvites;
}

public
```

```
function addEmailInvites( ):void
{
for (var key:Object in _emailInvites)
{
if (!(_emailInvites[key] in _invitedAlliances))
{
var alliance:AllianceVO = _alliances[key];
if(alliance)
{
// inviterKey and _inviterName are unused now so let's leave it empty so we won't have to fetch
this data
var allianceInvite:AllianceInviteVO = new AllianceInviteVO("","",alliance);
addInvitedAlliance(allianceInvite);
}
}
}
}

public function getAlliances():Dictionary { return _alliances; }
}
}

-------------------------------------
File 290: igw\com\model\alliance\AllianceVO.as
package com.model.alliance
{
import com.model.prototype.IPrototype;

public class AllianceVO
{
private var _key:String;
private var _name:String;
private var _faction:IPrototype;
private var _motd:String;
private var _description:String;
private var _isPublic:Boolean;
private var _memberCount:int;
private var _members:Vector.<AllianceMemberVO>;

public function AllianceVO( key:String, name:String, faction:IPrototype, motd:String,
description:String, isPublic:Boolean )
{
_key = key;
_name = name;
_faction = faction;
_motd = motd;
_description = description;
_isPublic = isPublic;
}

public
```

```
function get key():String { return _key; }
public function get name():String { return _name; }
public function get faction():IPrototype { return _faction; }
public function get motd():String { return _motd; }
public function get description():String { return _description; }
public function get isPublic():Boolean { return _isPublic; }
public function set memberCount( v:int ):void { _memberCount = v; }
public function get memberCount():int { return _memberCount; }
public function set members( v:Vector.<AllianceMemberVO> ):void
{
_members = v;
if (_members && _members.length > 0)
_memberCount = _members.length;
}
public function get members():Vector.<AllianceMemberVO> { return _members; }
}
}

-------------------------------------
File 291: igw\com\model\asset\AssetModel.as
package com.model.asset
{
import com.enum.CategoryEnum;
import com.event.RequestLoadEvent;
import com.game.entity.components.shared.Detail;
import com.model.Model;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.service.loading.LoadPriority;
import com.service.loading.LoadingTypes;

import flash.display3D.Context3DTextureFormat;
import flash.utils.Dictionary;
import flash.utils.getTimer;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.ash.core.Entity;
import org.ash.tick.ITickProvider;
import org.swiftsuspenders.Injector;

public class AssetModel extends Model
{
public static var instance:AssetModel;

public static const FAILED:String = "failed";
protected static const LOADING:String = "loading";
protected const _logger:ILogger = getLogger('AssetModel');

protected var _audioProtos:Vector.<IPrototype>;
protected
```

```
var _cache:Dictionary;
protected var _callbacks:Dictionary;
protected var _frameTick:ITickProvider;
protected var _injector:Injector;
protected var _spritePacks:Dictionary;
protected var _spriteSheets:Dictionary;
protected var _spriteSheetBuilds:Vector.<ISpriteSheet>;

public function AssetModel()
{
_audioProtos = new Vector.<IPrototype>;
_cache = new Dictionary();
_callbacks = new Dictionary();
_spritePacks = new Dictionary();
_spriteSheets = new Dictionary();
_spriteSheetBuilds = new Vector.<ISpriteSheet>;
instance = this;
}

public function getAssetVOByName(key:String):AssetVO{
return _cache[key];
}

public function cache( url:String, asset:Object ):void
{
if (_cache[url] == null)
_cache[url] = asset;
else if (_cache[url] == LOADING)
{
_cache[url] = asset;
applyCallbacks(asset, url);
}
}

public function getFromCache( url:String, callback:Function = null, priority:int =
3/*LoadPriority.LOW*/, absoluteURL:Boolean = false ):Object
{
if (_cache[url] == null)
{
dispatch(new RequestLoadEvent(url, priority, absoluteURL));
_cache[url] = LOADING;
}
if (_cache[url] == LOADING)
{
if (callback != null)
addCallback(callback, url);
return null;
}
if (callback != null)
callback.apply(this, [_cache[url]]);
return
```

```
_cache[url];
}

public function removeFromCache( url:String ):void
{
_cache[url] = null;
delete _cache[url];
}

public function clearCache():void
{
for (var url:String in _cache)
removeFromCache(url);
}

public function addGameAssetData( data:IPrototype ):void
{
if (!_cache[data.name])
_cache[data.name] = new AssetVO();
_cache[data.name].addGameAssetData(data);
}

public function addUIAssetData( data:IPrototype ):void
{
if (!_cache[data.name])
_cache[data.name] = new AssetVO();
_cache[data.name].addUIAssetData(data);
}

public function addAudioAssetData( data:IPrototype ):void
{
if (!_cache[data.name])
_cache[data.name] = new AssetVO();
_cache[data.name].addAudioAssetData(data);

_audioProtos.push(data);
}

public function addFilterAssetData( data:IPrototype ):void
{
if (!_cache[data.name])
_cache[data.name] = new AssetVO();
_cache[data.name].addFilterAssetData(data);
}

public function getAudioProtos():Vector.<IPrototype> { return _audioProtos; }

public function removeGameAssetData( name:String ):void
{
if (_cache[name])
{
```

```
_cache[name] = null;
delete _cache[name];
}
}

public function getEntityData( type:String ):AssetVO
{
if (_cache[type])
return _cache[type];
else
_logger.debug('No entity data found for: {0}', [type]);
return null;
}

public function getSpritePack( type:String, load:Boolean = true, entity:Entity = null, priority:int = 3
/*LoadPriority.LOW*/, format:String = Context3DTextureFormat.BGRA ):ISpritePack
{
if (_spritePacks[type])
{
return _spritePacks[type];
}
if (!_cache[type])
{
_logger.debug('No entity data found for: {0}', [type]);
return null;
} else
{
if (_spritePacks[type] == null)
{
//check to see if any of these sprite sheets have already been loaded
var detail:Detail;
var vo:AssetVO = _cache[type];
var toLoad:Array = [];
var sheet:ISpriteSheet;
_spritePacks[type] = new SpritePack(type, vo.usedBy, format);
for (var i:int = 0; i < vo.sprites.length; i++)
{
sheet = getSpriteSheet(vo.sprites[i], vo.isMesh);
if (!sheet.begunLoad)
{
if (load)
{
if (vo.isMesh)
toLoad.push(vo.sprites[i]);
else
{
toLoad.push(vo.sprites[i]);
toLoad.push(vo.spriteXML[i]);
}
sheet.begunLoad
```

```
= true;

//find the priority if we have an entity
if (entity)
{
detail = entity.get(Detail);
switch (detail.category)
{
case CategoryEnum.BUILDING:
priority = LoadPriority.HIGH;
break;
case CategoryEnum.SECTOR:
priority = LoadPriority.HIGH;
break;
case CategoryEnum.SHIP:
priority = detail.prototypeVO.getValue("faction") == CurrentUser.faction ?
LoadPriority.IMMEDIATE : LoadPriority.MEDIUM;
break;
case CategoryEnum.STARBASE:
priority = LoadPriority.IMMEDIATE;
break;
}
}
} else
sheet.begunLoad = true;
}
_spritePacks[type].addSpriteSheet(sheet);
}
if (toLoad.length > 0)
{
var event:RequestLoadEvent = new RequestLoadEvent(null, priority);
event.batchLoad(vo.isMesh ? LoadingTypes.SPRITE_SHEET_MESH :
LoadingTypes.SPRITE_SHEET, toLoad);
dispatch(event);
}
}
}
return _spritePacks[type];
}

public function initSpriteSheet( url:String, asset:*, xml:XML, build:Boolean = true ):void
{
if (_spriteSheets[url])
{
_spriteSheets[url].init(asset, xml, url);
if (build)
{
//trace(url, _spriteSheets[url].referenceCount);
_spriteSheetBuilds.push(_spriteSheets[url]);
if (_spriteSheetBuilds.length == 1)
_frameTick.addFrameListener(buildSpriteSheets);
```

```
}
}
}

public function getSpriteSheet( url:String, isMesh:Boolean = false ):ISpriteSheet
{
if (!_spriteSheets[url])
_spriteSheets[url] = /*(isMesh) ? new MeshSheet() : */ _injector.getInstance(ISpriteSheet);
return _spriteSheets[url];
}

public function removeSpritePack( pack:ISpritePack ):void
{
if (!pack)
return;
_spritePacks[pack.type] = null;
delete _spritePacks[pack.type];
for (var i:int = 0; i < pack.spriteSheets.length; i++)
{
pack.spriteSheets[i].decReferenceCount();
if (pack.spriteSheets[i].referenceCount <= 0)
{
//trace("removing", pack.type, pack.spriteSheets[i].url, pack.spriteSheets[i].referenceCount);
removeSpriteSheet(pack.spriteSheets[i].url);
}
}
pack.destroy();
}

public function removeAllSpritePacks():void
{
for each (var pack:ISpritePack in _spritePacks)
{
removeSpritePack(pack);
}
}

public function removeSpriteSheet( url:String ):void
{
if (!_spriteSheets[url])
return;
if (_spriteSheets[url] == null)
return;
_spriteSheets[url].destroy();
_spriteSheets[url] = null;
delete _spriteSheets[url];
}

public function stopAllSpritePackBuilds():void
{
```

```
_spriteSheetBuilds.length = 0;
_frameTick.removeFrameListener(buildSpriteSheets);
}

private function buildSpriteSheets( time:Number ):void
{
var endBuild:Number = getTimer() + 20;
while (getTimer() < endBuild && _spriteSheetBuilds.length > 0)
{
if (!_spriteSheetBuilds[0].build() || _spriteSheetBuilds[0].built)
{
_spriteSheetBuilds.shift();
//stop the build loop if there is nothing left to build
if (_spriteSheetBuilds.length == 0)
_frameTick.removeFrameListener(buildSpriteSheets);
}
}
}

private function addCallback( callback:Function, url:String ):void
{
if (callback != null && url)
{
if (!_callbacks[url])
_callbacks[url] = [];
if (_callbacks[url].indexOf(callback) == -1)
_callbacks[url].push(callback);
}
}

private function applyCallbacks( asset:Object, url:String ):void
{
if (asset && url)
{
if (_callbacks.hasOwnProperty(url))
{
var assetArg:Array = [asset];
var callbacks:Array = _callbacks[url];
for (var i:int = 0; i < callbacks.length; i++)
{
callbacks[i].apply(null, assetArg);
}

//remove the callbacks
_callbacks[url] = null;
delete _callbacks[url];
}
}
}

[Inject]
```

```
public function set frameTick( v:ITickProvider ):void { _frameTick = v; }
[Inject]
public function set injector( v:Injector ):void { _injector = v; }

public function get spritePacks():Dictionary { return _spritePacks; }
}
}

-------------------------------------
File 292: igw\com\model\asset\AssetVO.as
package com.model.asset
{
import com.model.prototype.IPrototype;

import flash.geom.Rectangle;

import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;

public class AssetVO
{
private var _audio:String;
private var _bbox:Rectangle;
private var _type:String;
private var _isMesh:Boolean = false;
private var _iconImage:String;
private var _smallImage:String;
private var _mediumImage:String;
private var _largeImage:String;
private var _profileImage:String;
private var _loops:int;
private var _descriptionText:String;
private var _radius:Number = 1;
private var _scale:Number = 1;
private var _spriteName:String;
private var _sprites:Array;
private var _spriteXML:Array;
private var _schematicLayoutData:Array;
private var _shieldScale:Number = 1;
private var _spriteSheetsString:String;
private var _spriteSheetsMobileString:String;
private var _usedBy:int;
private var _visibleName:String;
private var _volume:Number;


private var _key:String;

//Filters
private
```

```
var _sort:Number;
private var _filterIcon:String;
private var _filterIconSelected:String;
private var _filterIconHover:String;

public function AssetVO()
{
_sprites = [];
_spriteXML = [];
_schematicLayoutData = [];
}

public function addGameAssetData( data:IPrototype ):void
{
_type = data.name;
_key = data.getUnsafeValue('key');
if (String(data.getValue('spriteSheets')).length > 0)
{
_spriteSheetsString = data.getValue("spriteSheets");
_spriteSheetsMobileString = data.getValue("spriteSheetsMobile");
var ss:Array = (DeviceMetrics.PLATFORM == PlatformEnum.MOBILE &&
_spriteSheetsMobileString != null && _spriteSheetsMobileString != '') ?
_spriteSheetsMobileString.split(',') :
_spriteSheetsString.split(',');
var ext:String = data.getUnsafeValue('jpg') == true ? '.jpg' : '.png';
for (var i:int = 0; i < ss.length; i++)
{
_sprites.push('sprite/' + ss[i] + ext);
_spriteXML.push('sprite/' + ss[i] + '.xml');
}
}
if (data.getUnsafeValue('bbox') != null && data.getUnsafeValue('bbox') != '')
{
var tmp:Array = String(data.getUnsafeValue('bbox')).split(',');
_bbox = new Rectangle(tmp[0], tmp[1], tmp[2], tmp[3]);
}
/*
var mesh:String = data.getUnsafeValue('mesh');
if (mesh != '' && mesh != null)
{
trace("made it");
_isMesh = true;
_spriteXML.length = 0;
_sprites.length = 0;
_sprites.push('mesh/' + mesh);
}*/
_spriteName = data.getUnsafeValue('spriteName');
_radius = data.getUnsafeValue('radius');
_scale = (data.getUnsafeValue('scale') != null) ? data.getUnsafeValue('scale') : 1;
_shieldScale = (data.getUnsafeValue('shieldScale') != null) ? data.getUnsafeValue('shieldScale')
:
```

```
1;
_usedBy = data.getUnsafeValue('usedBy');
}

public function addUIAssetData( data:IPrototype ):void
{
_type = data.name;
_iconImage = data.getValue('imageIcon');
_smallImage = data.getUnsafeValue('imageSmall');
_mediumImage = data.getUnsafeValue('imageMedium');
_largeImage = data.getValue('imageLarge');
_profileImage = data.getValue('imageProfile');
_visibleName = data.getValue('localizedUIName');
_descriptionText = data.getValue('localizedDescriptionText');
}

public function addAudioAssetData( data:IPrototype ):void
{
_audio = data.getValue('audio');
if (_audio == "")
_audio = null;
_loops = data.getValue('loops');
_volume = data.getValue('volume');
}

public function addFilterAssetData( data:IPrototype ):void
{
_type = data.name;
_sort = data.getValue('sort');
_visibleName = data.getValue('uiName');
_filterIcon = data.getValue('icon');
_filterIconSelected = data.getValue('iconSelected');
_filterIconHover = data.getValue('iconHover');
}

public function setOneSpriteXML(value:String):void
{
while( _spriteXML.length >0)
_spriteXML.pop();

_spriteXML.push(value);
}
public function setSpriteXML(id:int, value:String):void { if(id < _spriteXML.length){_spriteXML[id]
= value;}}
public function set spriteSheetsString(value:String):void { _spriteSheetsString =value;}

public function get iconImage():String { return _iconImage; }
public function get smallImage():String { return _smallImage; }
public function get mediumImage():String { return _mediumImage; }
public function get largeImage():String { return _largeImage; }
public
```

```
function get profileImage():String { return _profileImage; }
public function get bbox():Rectangle { return _bbox; }
public function get type():String { return _type; }
public function get radius():Number { return _radius; }
public function get scale():Number { return _scale; }
public function get isMesh():Boolean { return _isMesh; }
public function get spriteName():String { return _spriteName; }
public function get sprites():Array { return _sprites; }
public function get spriteXML():Array { return _spriteXML; }
public function get spriteSheetsString():String { return _spriteSheetsString; }
public function get schematicLayoutData():Array { return _schematicLayoutData; }
public function get visibleName():String { return _visibleName; }
public function get shieldScale():Number { return _shieldScale; }
public function get descriptionText():String { return _descriptionText; }
public function get usedBy():int { return _usedBy; }

public function get key():String { return _key; }

//audio
public function get audio():String { return _audio; }
public function get loops():int { return _loops; }
public function get volume():Number { return _volume; }

//filter
public function get sort():Number { return _sort; }
public function get filterIcon():String { return _filterIcon; }
public function get filterIconSelected():String { return _filterIconSelected; }
public function get FilterIconHover():String { return _filterIconHover; }

}
}

-----------------------------------
File 293: igw\com\model\asset\ISpritePack.as
package com.model.asset
{
public interface ISpritePack
{
function getFrame( label:String, frame:int ):*;

function getFrames( label:String ):Array;

function addSpriteSheet( sheet:ISpriteSheet ):void;

function get is3D():Boolean;
function get ready():Boolean;
function get spriteSheets():Vector.<ISpriteSheet>;
function get type():String;
function get usedBy():int;

function
```

```
destroy():void;
}
}

------------------------------------
File 294: igw\com\model\asset\ISpriteSheet.as
package com.model.asset
{
public interface ISpriteSheet
{
function init( sprite:*, xml:XML, url:String ):void;

function getFrame( label:String, frame:int ):*;

function getFrames( label:String ):Array;

function build():Boolean;

function incReferenceCount():void;
function decReferenceCount():void;

function get built():Boolean;
function get begunLoad():Boolean;
function set begunLoad( v:Boolean ):void;
function set format( v:String ):void;
function get is3D():Boolean;
function get referenceCount():int;
function get url():String;

function destroy():void;
}
}

------------------------------------
File 295: igw\com\model\asset\MeshSheet.as
package com.model.asset
{
import flash.utils.Dictionary;

import org.away3d.entities.Mesh;

public class MeshSheet extends SpriteSheetBase implements ISpriteSheet
{
private var _height:Number;
private var _mesh:Mesh;
private var _width:Number;

override public function init( sprite:*, xml:XML, url:String ):void
{
_built = true;
_subtextureIndex
```

```
= 0;
_frames = new Dictionary();
_mesh = sprite;
_xml = (xml != null) ? xml.SubTexture : null;
_url = url;
}

override public function getFrame( label:String, frame:int ):*
{
return _mesh;
}

override public function getFrames( label:String ):Array
{
if (!_frames[label])
_frames[label] = [_mesh];
return _frames[label];
}

override public function get is3D():Boolean { return true; }

override public function destroy():void
{
/*if (_frames)
{
for (var label:String in _frames)
{
for (var i:int = 0; i < _frames[label].length; i++)
{
_frames[label][i].dispose();
}
_frames[label].length = 0;
_frames[label] = null;
delete _frames[label];
}
}*/
super.destroy();
}
}
}

-----------------------------------
File 296: igw\com\model\asset\SpritePack.as
package com.model.asset
{
import flash.utils.Dictionary;

public class SpritePack implements ISpritePack
{
private var _cache:Dictionary;
private
```

```
var _format:String;
private var _is3D:Boolean;
private var _ready:Boolean;
private var _spriteSheets:Vector.<ISpriteSheet>;
private var _type:String;
private var _usedBy:int;

public function SpritePack( type:String, usedBy:int, format:String )
{
_cache = new Dictionary(true);
_format = format;
_is3D = _ready = false;
_spriteSheets = new Vector.<ISpriteSheet>;
_type = type;
_usedBy = usedBy;
}

public function getFrame( label:String, frame:int ):*
{
if (!_ready)
return null;
if (!_cache[label])
getFrames(label);
return _cache[label][frame];
}

public function getFrames( label:String ):Array
{
if (!_ready)
return null;
if (!_cache[label])
{
for (var i:int = 0; i < _spriteSheets.length; i++)
{
if (_spriteSheets[i].getFrames(label))
{
_cache[label] = _spriteSheets[i].getFrames(label);
break;
}
}
}
return _cache[label];
}

public function addSpriteSheet( sheet:ISpriteSheet ):void
{
if (!sheet.built)
_ready = false;
if (!_is3D && sheet.is3D)
_is3D = true;
sheet.format
```

```
= _format;
sheet.incReferenceCount();
_spriteSheets.push(sheet);
}

public function get is3D():Boolean { return _is3D; }
public function get ready():Boolean
{
if (!_ready && _spriteSheets.length > 0)
{
for (var i:int = 0; i < _spriteSheets.length; i++)
{
if (!_spriteSheets[i].built)
return false;
}
_ready = true;
}
return true;
}

public function get spriteSheets():Vector.<ISpriteSheet> { return _spriteSheets; }
public function get type():String { return _type; }
public function get usedBy():int { return _usedBy; }

public function destroy():void
{
_cache = null;
_ready = false;
_spriteSheets.length = 0;
_spriteSheets = null;
}
}
}

-----------------------------------
File 297: igw\com\model\asset\SpriteSheet.as
package com.model.asset
{
import flash.display.BitmapData;
import flash.geom.Point;
import flash.geom.Rectangle;

public class SpriteSheet extends SpriteSheetBase
{
private var _dest:Point;
private var _height:Number;
private var _width:Number;

override public function init( sprite:*, xml:XML, url:String ):void
{
_dest
```

```
= new Point();
super.init(sprite, xml, url);
}

override protected function cutFrame( label:String, frame:int, region:Rectangle,
frameRect:Rectangle ):void
{
_width = (!frameRect) ? region.width : frameRect.width;
_height = (!frameRect) ? region.height : frameRect.height;
_dest.setTo((!frameRect) ? 0 : Math.abs(frameRect.x), (!frameRect) ? 0 :
Math.abs(frameRect.y));
var bmd:BitmapData = new BitmapData(_width, _height, true, 0);
bmd.copyPixels(_sprite, region, _dest);
_frames[label][frame] = bmd;
}

public function addFrame( label:String, frame:int, bmd:BitmapData, forceBuilt:Boolean = false
):void
{
if (!_frames[label])
_frames[label] = [];
_frames[label][frame] = bmd;

if (forceBuilt)
{
_begunLoad = _built = true;
_sprite = null;
cleanupBuild();
}
}

override public function destroy():void
{
if (_frames)
{
for (var label:String in _frames)
{
for (var i:int = 0; i < _frames[label].length; i++)
{
_frames[label][i].dispose();
}
_frames[label].length = 0;
_frames[label] = null;
delete _frames[label];
}
}
super.destroy();
}
}
}

-----------------------------------
```

File 298: igw\com\model\asset\SpriteSheetBase.as

```
package com.model.asset
{

import com.enum.TimeLogEnum;
import com.util.TimeLog;

import flash.display.BitmapData;
import flash.geom.Rectangle;
import flash.utils.Dictionary;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class SpriteSheetBase implements ISpriteSheet
{
protected const _logger:ILogger = getLogger('SpriteSheet');

protected var _begunLoad:Boolean = false;
protected var _built:Boolean = false;
protected var _format:String;
protected var _frames:Dictionary;
protected var _referenceCount:int = 0;
protected var _sprite:BitmapData;
protected var _subtextureIndex:int;
protected var _url:String;
protected var _xml:XMLList;

public function SpriteSheetBase()
{
_begunLoad = _built = false;
_referenceCount = 0;
}

public function init( sprite:*, xml:XML, url:String ):void
{
TimeLog.startTimeLog(TimeLogEnum.SPRITESHEET_PARSE, url);
_subtextureIndex = 0;
_frames = new Dictionary();
_sprite = BitmapData(sprite);
_xml = (xml != null) ? xml.SubTexture : null;
_url = url;
}

public function getFrame( label:String, frame:int ):*
{
if (!_frames[label])
{
_logger.debug('Label {} does not yet exist on the spritesheet.', label);
return
```

```
null;
}
if (frame > _frames[label].length)
frame = _frames[label].length - 1;
if (!_frames[label][frame])
{
_logger.debug('Frame {0} does not yet exist on the spritesheet for label {1}.', [frame, label]);
return null;
}

return _frames[label][frame];
}

public function getFrames( label:String ):Array
{
if (!_frames[label])
{
_logger.debug('Label {} does not yet exist on the spritesheet.', label);
return null;
}
return _frames[label];
}

public function build():Boolean
{
if (_xml == null)
return false;
var subTexture:XML = _xml[_subtextureIndex];
var name:String = subTexture.attribute("name");
var x:Number = parseFloat(subTexture.attribute("x"));
var y:Number = parseFloat(subTexture.attribute("y"));
var width:Number = parseFloat(subTexture.attribute("width"));
var height:Number = parseFloat(subTexture.attribute("height"));
var frameX:Number = parseFloat(subTexture.attribute("frameX"));
var frameY:Number = parseFloat(subTexture.attribute("frameY"));
var frameWidth:Number = parseFloat(subTexture.attribute("frameWidth"));
var frameHeight:Number = parseFloat(subTexture.attribute("frameHeight"));

var nparams:Array = name.split("^");
if (_frames[nparams[0]] == null)
_frames[nparams[0]] = [];

var region:Rectangle = new Rectangle(x, y, width, height);
var frame:Rectangle = frameWidth > 0 && frameHeight > 0 ?
new Rectangle(frameX, frameY, frameWidth, frameHeight) : null;
if (nparams.length > 1)
cutFrame(nparams[0], int(nparams[1]), region, frame);
else
cutFrame(nparams[0], 0, region, frame);

_subtextureIndex++;
```

```
if (_subtextureIndex == _xml.length())
{
cleanupBuild();
_built = true;
_subtextureIndex = 0;
}
return true;
}

protected function cutFrame( label:String, frame:int, region:Rectangle, frameRect:Rectangle
):void
{

}

protected function cleanupBuild():void
{
//dispose of the xml and the bitmapdatas that were used to build the spritesheet
_xml = null;
if (_sprite)
{
_sprite.dispose();
_sprite = null;
}
TimeLog.endTimeLog(TimeLogEnum.SPRITESHEET_PARSE, url);
}

public function incReferenceCount():void { _referenceCount++; }
public function decReferenceCount():void { _referenceCount--; }

public function get built():Boolean { return _built; }
public function get begunLoad():Boolean { return _begunLoad; }
public function set begunLoad( v:Boolean ):void { _begunLoad = v; }
public function set format( v:String ):void { _format = v; }
public function get is3D():Boolean { return false; }
public function get referenceCount():int { return _referenceCount; }
public function get url():String { return _url; }

public function destroy():void
{
_frames = null;
_referenceCount = 0;
_url = null;
_xml = null;
}
}
}

-----------------------------------
File
```

299: igw\com\model\asset\SpriteSheetStarling.as

```
package com.model.asset
{
import flash.geom.Rectangle;

import org.starling.textures.Texture;

public class SpriteSheetStarling extends SpriteSheetBase
{
private var _baseTexture:Texture;

override public function init( sprite:*, xml:XML, url:String ):void
{
super.init(sprite, xml, url);
//TODO crashes sometimes
_baseTexture = Texture.fromBitmapData(_sprite, false, false, 1, _format);
}

override protected function cutFrame( label:String, frame:int, region:Rectangle,
frameRect:Rectangle ):void
{
_frames[label][frame] = Texture.fromTexture(_baseTexture, region, frameRect);
}

override public function destroy():void
{
for (var label:String in _frames)
{
_frames[label].length = 0;
_frames[label] = null;
delete _frames[label];
}

if (_baseTexture)
_baseTexture.dispose();

_baseTexture = null;
super.destroy();
}
}
}
```

File 300: igw\com\model\battle\BattleEntityVO.as

```
package com.model.battle
{
import com.model.prototype.IPrototype;

public class BattleEntityVO
{
public
```

```
var category:String;
public var healthPercent:Number;
public var id:String;
public var ownerID:String;
public var prototype:IPrototype;
}
}

-----------------------------------
File 301: igw\com\model\battle\BattleModel.as
package com.model.battle
{
import com.controller.ServerController;
import com.enum.CategoryEnum;
import com.enum.server.BattleEntityTypeEnum;
import com.event.BattleEvent;
import com.model.Model;
import com.service.server.incoming.data.BattleEntityData;
import com.service.server.incoming.starbase.StarbaseAvailableRerollResponse;
import com.service.server.incoming.starbase.StarbaseRerollChanceResultResponse;
import com.service.server.incoming.starbase.StarbaseRerollReceivedResultResponse;
import com.service.server.incoming.starbase.StarbaseUnavailableRerollResponse;

import flash.events.Event;
import flash.geom.Point;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;


public class BattleModel extends Model
{
public var onRerollAdded:Signal;
public var onRerollUpdated:Signal;

public var onParticipantsAdded:Signal;

public var alloy:uint = 0;
public var credits:uint = 0;
public var energy:uint = 0;
public var synthetic:uint = 0;

public var baseFactionColor:uint;
public var baseOwnerID:String;
public var battleEndTick:int;
public var battleStartTick:int;
public var battleServerAddress:String;
public var isReplay:Boolean;
public var finished:Boolean;
public var focusLocation:Point = new Point();
public
```

```
var isBaseCombat:Boolean;

public var mapSizeX:int;
public var mapSizeY:int;

public var galacticName:String;
public var backgroundId:int;
public var planetId:int;
public var moonQuantity:int;
public var asteroidQuantity:int;
public var appearanceSeed:int;

public var isInstancedMission:Boolean;
public var missionID:String;
public var oldGameState:String;
public var oldSector:String;
public var participants:Vector.<String> = new Vector.<String>;
public var participantRatings:Dictionary = new Dictionary;
public var wonLastBattle:Boolean;

private var _availableRerolls:Dictionary;
private var _unavailableReroll:Dictionary;
private var _battleEntities:Vector.<BattleEntityVO> = new Vector.<BattleEntityVO>;
private var _battleEntityLookup:Dictionary;
private var _currentTick:int;

[PostConstruct]
public function init():void
{
galacticName = "";

_availableRerolls = new Dictionary();
_unavailableReroll = new Dictionary();
onRerollAdded = new Signal(BattleRerollVO);
onRerollUpdated = new Signal(BattleRerollVO);
onParticipantsAdded = new Signal(String);
}

public function addBattleEntity( data:BattleEntityData ):void
{
var vo:BattleEntityVO = new BattleEntityVO();
vo.category = (data.type == BattleEntityTypeEnum.SHIP) ? CategoryEnum.SHIP :
CategoryEnum.BUILDING;
vo.healthPercent = data.currentHealth / data.maxHealth;
vo.id = data.id;
vo.ownerID = data.ownerId;
vo.prototype = (data.type == BattleEntityTypeEnum.SHIP) ? data.shipPrototype :
data.buildingPrototype;
_battleEntities.push(vo);
if (_battleEntityLookup == null)
_battleEntityLookup
```

```
= new Dictionary(true);
_battleEntityLookup[data.id] = vo;
}

public function getBattleEntitiesByPlayer( playerID:String, category:String =
CategoryEnum.SHIP ):Vector.<BattleEntityVO>
{
var entities:Vector.<BattleEntityVO> = new Vector.<BattleEntityVO>;
for (var i:int = 0; i < _battleEntities.length; i++)
{
if (_battleEntities[i].ownerID == playerID && _battleEntities[i].category == category)
entities.push(_battleEntities[i]);
}
return entities;
}

public function getBattleEntity( id:String ):BattleEntityVO { return _battleEntityLookup[id]; }

public function getParticipantRating( id:String ):int
{
if (participantRatings.hasOwnProperty(id))
return participantRatings[id];
return 1;
}

public function addAvailableReroll( v:StarbaseAvailableRerollResponse ):void
{
var battleReroll:BattleRerollVO = new BattleRerollVO(v.battleKey, v.blueprintPrototype,
v.timeoutDelta);
_availableRerolls[v.battleKey] = battleReroll;
onRerollAdded.dispatch(battleReroll);
}

public function addUnavailableReroll( v:StarbaseUnavailableRerollResponse ):void
{
_unavailableReroll[v.battleKey] = v.reason;
}

public function updateRerollFromScan( v:StarbaseRerollChanceResultResponse ):void
{
if (v.battleKey in _availableRerolls)
{
var currentAvailableReroll:BattleRerollVO = _availableRerolls[v.battleKey];
currentAvailableReroll.scanned(v.blueprintPrototype, v.alloyReward, v.creditsReward,
v.energyReward, v.syntheticReward);
onRerollUpdated.dispatch(currentAvailableReroll);
}
}

public function updateRerollFromReroll( v:StarbaseRerollReceivedResultResponse ):void
{
```

```
if (v.battleKey in _availableRerolls)
{
var currentAvailableReroll:BattleRerollVO = _availableRerolls[v.battleKey];
currentAvailableReroll.rerolled(v.blueprintPrototype);
onRerollUpdated.dispatch(currentAvailableReroll);
}
}

public function removeRerollByID( id:String ):void
{
if (id in _availableRerolls)
delete _availableRerolls[id];
}

public function getAvailableRerollByID( id:String ):BattleRerollVO
{
if (id in _availableRerolls)
{
var currentAvailableReroll:BattleRerollVO = _availableRerolls[id];
if (currentAvailableReroll.timeRemaining > 0)
return currentAvailableReroll;
else
{
delete _availableRerolls[id];
return null
}
}

return null;
}

public function getUnavailableRerollByID( id:String ):int
{
if (id in _unavailableReroll)
{
var reroll:int = _unavailableReroll[id];
delete _unavailableReroll[id];
return reroll;
}
return 0;
}

public function getAllAvailableRerolls():Vector.<BattleRerollVO>
{
var battleRerolls:Vector.<BattleRerollVO> = new Vector.<BattleRerollVO>;
var currentAvailableReroll:BattleRerollVO;
for (var key:String in _availableRerolls)
{
currentAvailableReroll = _availableRerolls[key];
if
```

```
(currentAvailableReroll.timeRemaining > 0 && !currentAvailableReroll.hasPaid)
battleRerolls.push(currentAvailableReroll);
else
delete _availableRerolls[key];
}
return battleRerolls;
}

public function get battleEntities():Vector.<BattleEntityVO> { return _battleEntities; }

public function addParticipant( participant:String ):void
{
onParticipantsAdded.dispatch(participant);
participants.push(participant);
}

public function reconnect():void
{
var event:Event;
event = new BattleEvent(BattleEvent.BATTLE_JOIN, battleServerAddress);
dispatch(event);
}

/**
* @return The time remaining in miliseconds
*/
public function get timeRemaining():int
{
_currentTick = (ServerController.SIMULATED_TICK < battleStartTick) ? battleStartTick :
ServerController.SIMULATED_TICK;
if (_currentTick > battleEndTick)
_currentTick = battleEndTick;
return (battleEndTick - _currentTick) * 100;
}

public function cleanup():void
{
_battleEntities.length = 0;
_battleEntityLookup = null;
focusLocation.setTo(0, 0);
participants.length = 0;
participantRatings.length = 0;
}
}
}
```

File 302: igw\com\model\battle\BattleRerollVO.as

```
package
```

```
com.model.battle
{
import flash.utils.getTimer;

public class BattleRerollVO
{
private var _battleKey:String;
private var _recievedBlueprintPrototype:String;
private var _timeRemaining:Number;
private var _isReroll:Boolean;
private var _hasPaid:Boolean;

//paid
private var _blueprintPrototype:String;
private var _alloyReward:Number;
private var _creditsReward:Number;
private var _energyReward:Number;
private var _syntheticReward:Number;

private var _clientTime:Number;

public function BattleRerollVO( battleKey:String, recievedBlueprintPrototype:String,
timeRemaining:Number )
{
_battleKey = battleKey;
_recievedBlueprintPrototype = recievedBlueprintPrototype;
_timeRemaining = timeRemaining;

if (_recievedBlueprintPrototype)
_isReroll = true;

_clientTime = getTimer();
}

public function rerolled( blueprintPrototype:String ):void
{
if (!_hasPaid)
{
_hasPaid = true;
_blueprintPrototype = blueprintPrototype;
}
}

public function scanned( blueprintPrototype:String, alloyReward:Number,
creditsReward:Number, energyReward:Number, syntheticReward:Number ):void
{
if (!_hasPaid)
{
_hasPaid = true;
_blueprintPrototype = blueprintPrototype;
_alloyReward
```

```
= alloyReward;
_creditsReward = creditsReward;
_energyReward = energyReward;
_syntheticReward = syntheticReward;
}
}

public function get battleKey():String { return _battleKey; }
public function get recievedBlueprintPrototype():String { return _recievedBlueprintPrototype; }
public function get isReroll():Boolean { return _isReroll; }

public function get hasPaid():Boolean { return _hasPaid; }

public function get blueprintPrototype():String { return _blueprintPrototype; }
public function get alloyReward():Number { return _alloyReward; }
public function get creditsReward():Number { return _creditsReward; }
public function get energyReward():Number { return _energyReward; }
public function get syntheticReward():Number { return _syntheticReward; }

public function get timeRemaining():Number
{
var timeRemaining:Number = _timeRemaining - (getTimer() - _clientTime);
if (timeRemaining < 0)
timeRemaining = 0;
return timeRemaining;
}
}
}

-------------------------------------
File 303: igw\com\model\battlelog\BattleLogBaseInfoVO.as
package com.model.battlelog
{
public class BattleLogBaseInfoVO
{
private var _health:Number;
private var _buildings:Vector.<BattleLogEntityInfoVO>;
private var _baseRating:int;

public function BattleLogBaseInfoVO( health:Number, baseRating:int)
{
_health = health;
_baseRating = baseRating;
}

public function addBuildings( v:Vector.<BattleLogEntityInfoVO> ):void
{
_buildings = v;
}

public
```

```
function get health():Number { return _health; }
public function get baseRatings():int { return _baseRating; }
}
}
------------------------------------
File 304: igw\com\model\battlelog\BattleLogEntityInfoVO.as
package com.model.battlelog
{
public class BattleLogEntityInfoVO
{
private var _protoName:String;
private var _health:Number;

public function BattleLogEntityInfoVO( protoName:String, health:Number)
{
_protoName = protoName
_health = health;
}

public function get protoName():String { return _protoName; }
public function get health():Number { return _health; }
}
}
------------------------------------
File 305: igw\com\model\battlelog\BattleLogFleetInfoVO.as
package com.model.battlelog
{
import flash.utils.Dictionary;

public class BattleLogFleetInfoVO
{
private var _name:String;
private var _ships:Vector.<BattleLogEntityInfoVO>;
private var _fleetRating:int;
private var _fleetHealth:Number;
private var _alloyGained:int;
private var _energyGained:int;
private var _syntheticGained:int;

public function BattleLogFleetInfoVO( name:String, fleetRaiting:int, fleetHealth:Number,
alloyGained:int, energyGained:int, syntheticGained:int)
{
_name = name;
_fleetRating = fleetRaiting;
_fleetHealth = fleetHealth;
_alloyGained = alloyGained;
_energyGained = energyGained;
_syntheticGained = syntheticGained;
}

public
```

```
function addShips( v:Vector.<BattleLogEntityInfoVO> ):void
{
_ships = v;
}

public function fleetRating():int { return _fleetRating; }
public function fleetHealth():Number { return _fleetHealth; }
public function alloyGained():int { return _alloyGained; }
public function energyGained():int { return _energyGained; }
public function syntheticGained():int { return _syntheticGained; }
public function get ships():Vector.<BattleLogEntityInfoVO> { return _ships; }

}
}
```

File 306: igw\com\model\battlelog\BattleLogModel.as

```
package com.model.battlelog
{
import com.model.Model;
import com.service.server.incoming.battlelog.BattleLogDetailsResponse;
import com.service.server.incoming.data.BattleLogPlayerSummaryInfo;
import com.service.server.incoming.data.BattleLogSummaryInfo;

import org.osflash.signals.Signal;

public class BattleLogModel extends Model
{
private var _battleLogs:Vector.<BattleLogVO>;

public var countUpdated:Signal;
public var battleLogListUpdated:Signal;
public var battleLogDetailUpdated:Signal;

[PostConstruct]
public function init():void
{
_battleLogs = new Vector.<BattleLogVO>;
battleLogListUpdated = new Signal(Vector.<BattleLogVO>);
battleLogDetailUpdated = new Signal(BattleLogVO);
}

public function addBattleLogList( v:Vector.<BattleLogSummaryInfo> ):void
{
var oldLogs:Vector.<BattleLogVO> = _battleLogs.concat();
_battleLogs = new Vector.<BattleLogVO>;

var len:uint = v.length;
var currentBattlelog:BattleLogVO;
var currentBattleData:BattleLogSummaryInfo;
for(var i:uint = 0; i < len; ++i)
{
```

```
currentBattleData = v[i];
currentBattlelog = getMailByKey(currentBattleData.battleKey, oldLogs);

if(currentBattlelog == null && currentBattleData.battleKey != '')
currentBattlelog = createBattleLog(currentBattleData);

if(currentBattlelog)
_battleLogs.push(currentBattlelog);
}
oldLogs.length = 0;
battleLogListUpdated.dispatch(_battleLogs);
}

private function createBattleLog( currentBattleData:BattleLogSummaryInfo ):BattleLogVO
{
var winners:Vector.<BattleLogPlayerInfoVO> = new Vector.<BattleLogPlayerInfoVO>;
var losers:Vector.<BattleLogPlayerInfoVO> = new Vector.<BattleLogPlayerInfoVO>;
var len:uint;
var i:uint;
var newBattleLog:BattleLogVO = new BattleLogVO(currentBattleData.battleKey,
currentBattleData.startTime, currentBattleData.endTime);
var currentBattleWinners:Vector.<BattleLogPlayerSummaryInfo> = currentBattleData.winners;
var currentBattleLosers:Vector.<BattleLogPlayerSummaryInfo> = currentBattleData.losers;
var currentPlayerSummary:BattleLogPlayerSummaryInfo;
var newPlayer:BattleLogPlayerInfoVO;
len = currentBattleWinners.length;
for(; i < len; ++i)
{
currentPlayerSummary = currentBattleWinners[i];
if(currentPlayerSummary)
{
newPlayer = new BattleLogPlayerInfoVO(currentPlayerSummary.name,
currentPlayerSummary.playerKey, currentPlayerSummary.race, currentPlayerSummary.faction,
currentPlayerSummary.rating, currentPlayerSummary.wasBase);
winners.push(newPlayer);
}
}
newBattleLog.setWinners(winners);
len = currentBattleLosers.length;
for(i = 0; i < len; ++i)
{
currentPlayerSummary = currentBattleLosers[i];
if(currentPlayerSummary)
{
newPlayer = new BattleLogPlayerInfoVO(currentPlayerSummary.name,
currentPlayerSummary.playerKey, currentPlayerSummary.race, currentPlayerSummary.faction,
currentPlayerSummary.rating, currentPlayerSummary.wasBase);
losers.push(newPlayer);
}
}
```

```
newBattleLog.setLosers(losers);
newBattleLog.setHasReplay(currentBattleData.hasReplay);

return newBattleLog;
}

public function addBattleLogDetail( v:BattleLogDetailsResponse ):void
{
var battleLog:BattleLogVO = getMailByKey(v.battleKey, _battleLogs);
if(battleLog)
{
battleLog.updateWithDetails(v);
battleLogDetailUpdated.dispatch(battleLog);
}
}

public function getMailByKey( battleKey:String, battleLogHolder:Vector.<BattleLogVO>
):BattleLogVO
{
var battleLog:BattleLogVO;
if(battleLogHolder)
{
var len:uint = battleLogHolder.length;
var currentBattleLog:BattleLogVO;
for(var i:uint = 0; i < len; ++i)
{
currentBattleLog = battleLogHolder[i];
if(currentBattleLog.battleKey == battleKey)
{
battleLog = currentBattleLog;
break;
}
}

}

return battleLog;
}
}
}
-------------------------------------
File 307: igw\com\model\battlelog\BattleLogPlayerInfoVO.as
package com.model.battlelog
{
import com.service.server.incoming.data.BattleLogBaseDetailInfo;
import com.service.server.incoming.data.BattleLogEntityDetailInfo;
import com.service.server.incoming.data.BattleLogFleetDetailInfo;

public class BattleLogPlayerInfoVO
{
```

```
private var _name:String;
private var _playerKey:String;
private var _race:String;
private var _faction:String;
private var _rating:int;
private var _wasBase:Boolean;

//Update
private var _hasFleet:Boolean;
private var _fleet:BattleLogFleetInfoVO;
private var _hasBase:Boolean;
private var _base:BattleLogBaseInfoVO;
private var _level:int;
private var _creditsGained:int;
private var _blueprintGained:String;

public function BattleLogPlayerInfoVO( name:String, playerKey:String, race:String,
faction:String, rating:int, wasBase:Boolean )
{
_name = name;
_playerKey = playerKey;
_race = race;
_faction = faction;
_rating = rating;
_wasBase = wasBase;
}

public function updatePlayerInfo( hasFleet:Boolean, fleet:BattleLogFleetDetailInfo,
hasBase:Boolean, base:BattleLogBaseDetailInfo, level:int, creditsGained:int,
blueprintGained:String ):void
{
_hasFleet = hasFleet;

if (fleet)
addFleet(fleet);

_hasBase = hasBase;

if (base)
addBase(base);

_level = level;
_creditsGained = creditsGained;
_blueprintGained = blueprintGained;
}

private function addFleet( fleet:BattleLogFleetDetailInfo ):void
{
var shipData:Vector.<BattleLogEntityDetailInfo> = fleet.ships;
var
```

```
len:uint = shipData.length;
var currentEntity:BattleLogEntityDetailInfo;
var newShip:BattleLogEntityInfoVO;
var ships:Vector.<BattleLogEntityInfoVO> = new Vector.<BattleLogEntityInfoVO>;
_fleet = new BattleLogFleetInfoVO(fleet.name, fleet.fleetRating, fleet.fleetHealth,
fleet.alloyGained, fleet.energyGained, fleet.syntheticGained);

for (var i:uint = 0; i < len; ++i)
{
currentEntity = shipData[i];
newShip = new BattleLogEntityInfoVO(currentEntity.protoName, currentEntity.health);
ships.push(newShip);
}
_fleet.addShips(ships);
}

private function addBase( base:BattleLogBaseDetailInfo ):void
{
var buildingData:Vector.<BattleLogEntityDetailInfo> = base.buildings;
var len:uint = buildingData.length;
var currentEntity:BattleLogEntityDetailInfo;
var newBuilding:BattleLogEntityInfoVO;
var buildings:Vector.<BattleLogEntityInfoVO> = new Vector.<BattleLogEntityInfoVO>;
_base = new BattleLogBaseInfoVO(base.baseHealth, base.baseRating);

for (var i:uint = 0; i < len; ++i)
{
currentEntity = buildingData[i];
newBuilding = new BattleLogEntityInfoVO(currentEntity.protoName, currentEntity.health);
buildings.push(newBuilding);
}
_base.addBuildings(buildings);
}

public function get name():String { return _name; }
public function get playerKey():String { return _playerKey; }
public function get race():String { return _race; }
public function get faction():String { return _faction; }
public function get rating():int { return _rating; }
public function get wasBase():Boolean { return _wasBase; }

public function get hasFleet():Boolean { return _hasFleet; }
public function get fleet():BattleLogFleetInfoVO { return _fleet; }
public function get hasBase():Boolean { return _hasBase; }
public function get base():BattleLogBaseInfoVO { return _base; }
public function get level():int { return _level; }
public function get creditsGained():int { return _creditsGained; }
public function get alloyGained():int { return (_hasFleet) ? _fleet.alloyGained() : 0; }
public function get energyGained():int { return (_hasFleet) ? _fleet.energyGained() : 0; }
public function get syntheticGained():int { return (_hasFleet) ? _fleet.syntheticGained() : 0; }
public
```

```
function get blueprintGained():String { return _blueprintGained; }
}
}

------------------------------------
File 308: igw\com\model\battlelog\BattleLogVO.as
package com.model.battlelog
{
import com.service.server.incoming.battlelog.BattleLogDetailsResponse;
import com.service.server.incoming.data.BattleLogPlayerDetailInfo;

public class BattleLogVO
{

private var _battleKey:String;
private var _winners:Vector.<BattleLogPlayerInfoVO>;
private var _losers:Vector.<BattleLogPlayerInfoVO>;
private var _startTime:Number;
private var _endTime:Number;
private var _timeSince:Number;
private var _hasDetails:Boolean;
private var _hasReplay:Boolean;

public function BattleLogVO( battleKey:String, startTime:Number, endTime:Number )
{
_winners = new Vector.<BattleLogPlayerInfoVO>;
_losers = new Vector.<BattleLogPlayerInfoVO>;
_battleKey = battleKey;
_startTime = startTime;
_endTime = endTime;
}

public function setWinners( winners:Vector.<BattleLogPlayerInfoVO> ):void
{
_winners = winners
}

public function setLosers( losers:Vector.<BattleLogPlayerInfoVO> ):void
{
_losers = losers;
}

public function setHasReplay( hasReplay:Boolean ):void
{
_hasReplay = hasReplay;
}

public function updateWithDetails( v:BattleLogDetailsResponse ):void
{
var winners:Vector.<BattleLogPlayerDetailInfo> = v.winners;
var
```

```
len:uint = winners.length;
var i:uint;
var currentPlayerData:BattleLogPlayerDetailInfo;
var currentPlayer:BattleLogPlayerInfoVO;
for (; i < len; ++i)
{
currentPlayerData = winners[i];
currentPlayer = getPlayerInfo(currentPlayerData.playerKey, _winners);
if (currentPlayer)
currentPlayer.updatePlayerInfo(currentPlayerData.hasFleet, currentPlayerData.fleet,
currentPlayerData.hasBase, currentPlayerData.base, currentPlayerData.level,
currentPlayerData.creditsGained,
currentPlayerData.blueprintGained);
}

var losers:Vector.<BattleLogPlayerDetailInfo> = v.losers;
len = losers.length;
for (i = 0; i < len; ++i)
{
currentPlayerData = losers[i];
currentPlayer = getPlayerInfo(currentPlayerData.playerKey, _losers);
if (currentPlayer)
currentPlayer.updatePlayerInfo(currentPlayerData.hasFleet, currentPlayerData.fleet,
currentPlayerData.hasBase, currentPlayerData.base, currentPlayerData.level,
currentPlayerData.creditsGained,
currentPlayerData.blueprintGained);
}

_hasDetails = true;
}

public function getPlayerInfo( id:String, v:Vector.<BattleLogPlayerInfoVO>
):BattleLogPlayerInfoVO
{
var len:uint = v.length;
var currentPlayer:BattleLogPlayerInfoVO;
for (var i:uint = 0; i < len; ++i)
{
currentPlayer = v[i];
if (currentPlayer.playerKey == id)
return currentPlayer;
}

return null;
}

public function get battleKey():String { return _battleKey; }
public function get winners():Vector.<BattleLogPlayerInfoVO> { return _winners; }
public function get losers():Vector.<BattleLogPlayerInfoVO> { return _losers; }
public function get startTime():Number { return _startTime; }
public
```

```
function get endTime():Number { return _endTime; }
public function get hasReplay():Boolean { return _hasReplay; }
public function get timeSince():Number { return (new Date()).time - endTime; }
public function get hasDetails():Boolean { return _hasDetails; }
}
}

-----------------------------------
File 309: igw\com\model\blueprint\BlueprintModel.as
package com.model.blueprint
{
import com.enum.ToastEnum;
import com.event.ToastEvent;
import com.model.Model;
import com.model.prototype.IPrototype;
import com.service.language.Localization;
import com.service.server.incoming.data.BlueprintData;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class BlueprintModel extends Model
{
private var _ownedBlueprints:Dictionary = new Dictionary();

private var _blueprintCompleteText:String = 'CodeString.Toast.BlueprintComplete'; //Blueprint
Complete!
private var _blueprintPartText:String = 'CodeString.Toast.BlueprintPartGained'; //Parts
Collected: [[Number.partsCollected]]/[[Number.partsTotal]]

public function importPlayerBlueprints( blueprintData:BlueprintData, displayIfNotInList:Boolean =
false ):void
{
var blueprintVO:BlueprintVO = ObjectPool.get(BlueprintVO);
blueprintVO.init(blueprintData.id);
blueprintVO.importData(blueprintData);

if (blueprintVO.name in _ownedBlueprints)
{
var currentBlueprint:BlueprintVO = _ownedBlueprints[blueprintVO.name];
if (currentBlueprint.partsRemaining != blueprintVO.partsRemaining)
popToast(blueprintVO);
} else if (blueprintVO.partsCollected >= blueprintVO.totalParts && displayIfNotInList)
popToast(blueprintVO);

_ownedBlueprints[blueprintVO.name] = blueprintVO;
}

public
```

```
function getBlueprintByName( name:String ):BlueprintVO
{
if (name in _ownedBlueprints)
return _ownedBlueprints[name]

return null;
}

public function getBlueprintByUIName( name:String ):BlueprintVO
{
for each (var vo:BlueprintVO in _ownedBlueprints)
{
var bpPrototype:IPrototype = vo.prototype;
if(bpPrototype.getValue('uiAsset') == name)
{
return vo;
}
}

return null;
}

public function getBlueprintByID( id:String ):BlueprintVO
{
for each (var vo:BlueprintVO in _ownedBlueprints)
{
if (vo.id == id)
return vo;
}

return null;
}

public function removeBlueprintByName( name:String ):void
{
if (name in _ownedBlueprints)
delete _ownedBlueprints[name];
}

public function get ownedBlueprints():Dictionary { return _ownedBlueprints; }

private function popToast( blueprint:BlueprintVO ):void
{
var text:String;
if (blueprint.complete)
text = Localization.instance.getString(_blueprintCompleteText);
else
text = Localization.instance.getStringWithTokens(_blueprintPartText,
{'[[Number.partsCollected]]':blueprint.partsCollected,
'[[Number.partsTotal]]':blueprint.totalParts});

var
```

```
toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = ToastEnum.BLUEPRINT;
toastEvent.prototype = blueprint;
if (text != '')
toastEvent.addStringsFromArray([text]);
dispatch(toastEvent);
}
}
}

------------------------------------
File 310: igw\com\model\blueprint\BlueprintVO.as
package com.model.blueprint
{
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.BlueprintData;

public class BlueprintVO implements IPrototype
{
private var _id:String;
private var _prototype:IPrototype;
private var _playerOwner:String;
private var _partsCollected:int;
private var _partsCollectedBank:int;
private var _blueprintPrototype:String;

public function init( id:String ):void
{
_id = id;
}

public function importData( blueprintData:BlueprintData ):void
{
_id = blueprintData.id
_prototype = blueprintData.prototype;
_playerOwner = blueprintData.playerOwner;
_partsCollected = blueprintData.partsCollected;
_partsCollectedBank = blueprintData.partsCollectedBank;
_blueprintPrototype = blueprintData.blueprintPrototype;
}
public function get partsCompleted():int
{
return _partsCollected;
}

public function get partsCollected():int
{
return _partsCollectedBank;
}

public
```

```
function get totalParts():int
{
return getUnsafeValue('parts');
}

public function get partsRemaining():int
{
return getUnsafeValue('parts') - _partsCollectedBank;
}

public function get complete():Boolean
{
return (getUnsafeValue('parts') - _partsCollected <= 0) ? true : false;
}

public function get costScale():Number
{
return getUnsafeValue('costScale');
}

public function get prototype():IPrototype { return _prototype; }

public function getUnsafeValue( key:String ):* { return _prototype.getUnsafeValue(key); }
public function getValue( key:String ):* { return _prototype.getValue(key); }

public function get id():String { return _id; }

public function get asset():String { return _prototype.asset; }
public function get uiAsset():String { return _prototype.uiAsset; }

public function get name():String { return _prototype.name; }
public function get itemClass():String { return _prototype.itemClass; }
public function get buildTimeSeconds():uint { return _prototype.buildTimeSeconds; }

public function get alloyCost():int { return 0; }
public function get creditsCost():int { return 0; }
public function get energyCost():int { return 0; }
public function get syntheticCost():int { return 0; }

public function destroy():void
{
_prototype = null;
}
}
}

------------------------------------
File 311: igw\com\model\chat\ChatChannelVO.as
package com.model.chat
{
public
```

```
class ChatChannelVO
{
private var _channelDisplayName:String;
private var _channelID:int;
private var _channelColor:int;
private var _playerSendable:Boolean;

public function ChatChannelVO( channelID:int, displayName:String, channelColor:uint,
playerSendable:Boolean)
{
_channelID = channelID;
_channelDisplayName = displayName;
_channelColor = channelColor;
_playerSendable = playerSendable;
}

public function get displayName():String { return _channelDisplayName; }
public function set channelID( channelId:int ):void { _channelID = channelId; }
public function get channelID():int { return _channelID; }
public function get channelColor():uint { return _channelColor; }
public function get playerSendable():Boolean { return _playerSendable; }
}
}
```

File 312: igw\com\model\chat\ChatModel.as

```
package com.model.chat
{
import com.enum.server.ChatChannelEnum;
import com.google.analytics.debug.Panel;
import com.model.Model;
import com.service.language.Localization;
import com.util.CommonFunctionUtil;

import flash.utils.Dictionary;

import org.adobe.utils.DictionaryUtil;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class ChatModel extends Model
{
public var chatSignal:Signal;
public var activeChannelUpdated:Signal;
public var onDefaultChannelUpdated:Signal;
public var onDefaultChannelOverriden:Signal;

private var _chatPanels:Vector.<ChatPanelVO>;
private var _defaultChannel:ChatChannelVO;
private var _activeChannels:Dictionary;
private var _channels:Dictionary;

private
```

```
var _muteList:Vector.<String>;
private var _blockList:Vector.<String>;


private var _translatedGroup:String;
private var _translatedGlobal:String;
private var _translatedSector:String;
private var _translatedAlliance:String;
private var _translatedMembers:String;
private var _translatedFaction:String;

private var _chatColorDefault:uint = 0xea8440;
private var _chatColorFaction:uint = 0xea8440;
private var _chatColorGlobal:uint = 0xf0f0f0;
private var _chatColorSector:uint = 0xf1c232;
private var _chatColorGroup:uint = 0xea8440; // same as faction but currently unused
private var _chatColorAlliance:uint = 0x79da62;
private var _chatColorMembers:uint = 0x00ffff;
private var _chatColorSystem:uint = 0xff0000;
private var _chatColorBroadcast:uint = 0x52ea2e;
private var _chatColorEvent:uint = 0xbf68fc;
private var _chatColorWhisper:uint = 0xd980d3;

private var _whisperTarget:String;
private var _replyTarget:String;

private var _message:String = "<font color='#[[HexNumber.Color]]'><a
href='event:NameLink.[[String.NameLink]]'><font
color='#[[FactionHexNumber.Color]]'>[[[String.PlayerName]]]</font></a>:
[[String.Message]]<br></font>";
private var _systemMessage:String = "<font
color='#[[HexNumber.Color]]'>[[String.Message]]</font>";
private var _defaultMessage:String = "<font
color='#[[HexNumber.Color]]'>[[[String.PlayerName]]]: [[String.Message]]<br></font>"
private var _global:String = 'CodeString.Chat.ChannelName.Global'; //Global
private var _sector:String = 'CodeString.Chat.ChannelName.Sector'; //Sector
private var _group:String = 'CodeString.Chat.ChannelName.Group'; //Group
private var _faction:String = 'CodeString.Chat.ChannelName.Faction'; //Faction
private var _alliance:String = 'CodeString.Chat.ChannelName.Alliance'; //Alliance
private var _members:String = 'CodeString.Chat.ChannelName.Members'; //Members
private var _centerspace:String = 'CodeString.Chat.ChannelName.CenterSpace'; //Center
Space

private var _sectorTabName:String = 'CodeString.Chat.Tab.SectorName'; //SECTOR
private var _allianceTabName:String = 'CodeString.Chat.Tab.AllianceName'; //ALLIANCE
private var _factionTabName:String = 'CodeString.Chat.Tab.FactionName'; //FACTION
private var _globalTabName:String = 'EN';//'CodeString.Chat.Tab.GlobalName'; //GLOBAL
default name EN
private var _membersTabName:String = 'CodeString.Chat.Tab.MembersName'; //MEMBERS

public
```

```
function ChatModel()
{
chatSignal = new Signal(uint, String, Boolean);
activeChannelUpdated = new Signal(Dictionary);
onDefaultChannelOverriden = new Signal(uint);
onDefaultChannelUpdated = new Signal(ChatChannelVO);

_whisperTarget = '';
_replyTarget = '';

_muteList = new Vector.<String>;
_blockList = new Vector.<String>;

_chatPanels = new Vector.<ChatPanelVO>;

_activeChannels = new Dictionary();
_channels = new Dictionary();

addChannel(ChatChannelEnum.GLOBAL);
addChannel(ChatChannelEnum.FACTION);
addChannel(ChatChannelEnum.SECTOR);
addChannel(ChatChannelEnum.ALLIANCE);
addChannel(ChatChannelEnum.MEMBERS);
addChannel(ChatChannelEnum.GROUP);
addChannel(ChatChannelEnum.WHISPER);
addChannel(ChatChannelEnum.SYSTEM, true);
addChannel(ChatChannelEnum.BROADCAST, true);
addChannel(ChatChannelEnum.EVENT, true);

addPanel(_globalTabName, 0, [ChatChannelEnum.GLOBAL]);
addPanel(_factionTabName, 1, [ChatChannelEnum.FACTION]);
addPanel(_sectorTabName, 2, [ChatChannelEnum.SECTOR,
ChatChannelEnum.BROADCAST, ChatChannelEnum.EVENT, ChatChannelEnum.SYSTEM,
ChatChannelEnum.WHISPER]);
addPanel(_allianceTabName, 3, [ChatChannelEnum.ALLIANCE]);
addPanel(_membersTabName, 4, [ChatChannelEnum.MEMBERS]);
}

public function getChannelColorFromId( channelID:int ):int
{
var channelColor:int = _chatColorDefault;

switch (channelID)
{
case ChatChannelEnum.GLOBAL:
channelColor = _chatColorGlobal;
break;
case ChatChannelEnum.SECTOR:
channelColor = _chatColorSector;
break;
case
```

```
ChatChannelEnum.GROUP:
channelColor = _chatColorGroup;
break;
case ChatChannelEnum.ALLIANCE:
channelColor = _chatColorAlliance;
break;
case ChatChannelEnum.MEMBERS:
channelColor = _chatColorMembers;
break;
case ChatChannelEnum.SYSTEM:
channelColor = _chatColorSystem;
break;
case ChatChannelEnum.BROADCAST:
channelColor = _chatColorBroadcast;
break;
case ChatChannelEnum.EVENT:
channelColor = _chatColorEvent;
break;
case ChatChannelEnum.WHISPER:
channelColor = _chatColorWhisper;
break;
case ChatChannelEnum.FACTION:
channelColor = _chatColorFaction;
break;
}

return channelColor;
}

public function updateChannelColor( channelID:int, chatChannelColor:uint ):void
{
switch (channelID)
{
case ChatChannelEnum.GLOBAL:
_chatColorGlobal = chatChannelColor;
break;
case ChatChannelEnum.SECTOR:
_chatColorSector = chatChannelColor;
break;
case ChatChannelEnum.GROUP:
_chatColorGroup = chatChannelColor;
break;
case ChatChannelEnum.ALLIANCE:
_chatColorAlliance = chatChannelColor;
break;
case ChatChannelEnum.MEMBERS:
_chatColorMembers = chatChannelColor;
break;
case ChatChannelEnum.SYSTEM:
_chatColorSystem = chatChannelColor;
break;
```

```
case ChatChannelEnum.BROADCAST:
_chatColorBroadcast = chatChannelColor;
break;
case ChatChannelEnum.EVENT:
_chatColorEvent = chatChannelColor;
break;
case ChatChannelEnum.WHISPER:
_chatColorWhisper = chatChannelColor;
break;
case ChatChannelEnum.FACTION:
_chatColorWhisper = chatChannelColor;
break;
}
}

public function getChannelFromId( channelId:int ):ChatChannelVO
{
var channelToReturn:ChatChannelVO;
if (channelId in _channels)
channelToReturn = _channels[channelId];

return channelToReturn;
}

public function isInChannel( channelId:int ):Boolean
{
var inChannel:Boolean;
if (channelId in _activeChannels)
inChannel = true;

return inChannel;
}

public function getChannelIdFromName( channelName:String ):int
{
if (_translatedGlobal == '' || _translatedGlobal == null)
_translatedGlobal = Localization.instance.getString(_global);

if (_translatedGroup == '' || _translatedGroup == null)
_translatedGroup = Localization.instance.getString(_group);

if (_translatedSector == '' || _translatedSector == null)
_translatedSector = Localization.instance.getString(_sector);

if (_translatedAlliance == '' || _translatedAlliance == null)
_translatedAlliance = Localization.instance.getString(_alliance);

if (_translatedMembers == '' || _translatedMembers == null)
_translatedMembers = Localization.instance.getString(_members);

if
```

```
(_translatedFaction == '' || _translatedFaction == null)
_translatedFaction = Localization.instance.getString(_faction);

var channelID:int = -1;
switch (channelName)
{
case 'global':
case _translatedGlobal:
channelID = ChatChannelEnum.GLOBAL;
break;
case 'sector':
case _translatedSector:
channelID = ChatChannelEnum.SECTOR;
break;
case 'group':
case _translatedGroup:
channelID = ChatChannelEnum.GROUP;
break;
case 'alliance':
case _translatedAlliance:
channelID = ChatChannelEnum.ALLIANCE;
break;
case 'members':
case _translatedMembers:
channelID = ChatChannelEnum.MEMBERS;
break;
case 'faction':
case _translatedFaction:
channelID = ChatChannelEnum.FACTION;
break;
}

return channelID;
}

public function addPanel( name:String, id:uint, channels:Array ):void
{
var newPanel:ChatPanelVO = new ChatPanelVO(name, id);
var len:uint = channels.length;
for (var i:uint = 0; i < len; ++i)
newPanel.addChannel(channels[i], getChannelFromId(channels[i]));

chatPanels.push(newPanel);
}

public function removePanel( id:uint ):void
{
var len:uint = chatPanels.length;
var currentPanel:ChatPanelVO;
for (var i:uint = 0; i < len; ++i)
{
```

```
currentPanel = chatPanels[i];
if (currentPanel.id == id)
{
currentPanel = null;
chatPanels.splice(i, 1);
break;
}

}
}

public function addChannelToPanel( panelID:uint, channelID:int ):void
{
var len:uint = chatPanels.length;
var currentPanel:ChatPanelVO;
for (var i:uint = 0; i < len; ++i)
{
currentPanel = chatPanels[i];
if (currentPanel.id == panelID)
{
currentPanel.addChannel(channelID, getChannelFromId(channelID));
break;
}

}
}

public function removeChannelToPanel( panelID:uint, channelID:int ):void
{
var len:uint = chatPanels.length;
var currentPanel:ChatPanelVO;
for (var i:uint = 0; i < len; ++i)
{
currentPanel = chatPanels[i];
if (currentPanel.id == panelID)
{
currentPanel.removeChannel(channelID);
break;
}

}
}

public function getPanelLogs( panelID:uint ):String
{
var len:uint = chatPanels.length;
var currentPanel:ChatPanelVO;
for (var i:uint = 0; i < len; ++i)
{
currentPanel
```

```
= chatPanels[i];
if (currentPanel.id == panelID)
{
return currentPanel.logs;
}

}
return '';
}

public function addChannel( channelId:int, isActive:Boolean = false ):void
{
var channelName:String = getChannelNameFromId(channelId);
var playerSendable:Boolean = getChannelPlayerSendableFromId(channelId);
var channelColor:int = getChannelColorFromId(channelId);

var channel:ChatChannelVO = new ChatChannelVO(channelId, channelName, channelColor,
playerSendable);

_channels[channelId] = channel;

if (isActive)
{
_activeChannels[channelId] = channel;

if (_defaultChannel == null && channelId == ChatChannelEnum.GLOBAL)
defaultChannel = channel;

activeChannelUpdated.dispatch(_activeChannels);
}
}

public function activateChannel( channelId:int ):void
{
if (channelId in _channels)
{
var channelVO:ChatChannelVO = _channels[channelId];

_activeChannels[channelId] = channelVO;

if (_defaultChannel == null && channelId == ChatChannelEnum.GLOBAL)
defaultChannel = channelVO;

activeChannelUpdated.dispatch(_activeChannels);
}
}

public function deactivateChannel( channelId:int ):void
{
if (channelId in _channels)
{
```

```
if (channelId in _activeChannels)
{
if (_defaultChannel == _activeChannels[channelId])
_defaultChannel = null;

delete _activeChannels[channelId];

if (ChatChannelEnum.SECTOR in _activeChannels)
_defaultChannel = _activeChannels[ChatChannelEnum.SECTOR];

activeChannelUpdated.dispatch(_activeChannels);
}
}
}

public function addChatLog( vo:ChatVO ):void
{
var currentPanel:ChatPanelVO;
var len:uint = _chatPanels.length;
for (var i:uint = 0; i < len; ++i)
{
currentPanel = _chatPanels[i];
if (currentPanel.listeningToChannel(vo.channelID))
{
currentPanel.addChatLog(vo, convertVOToText(vo));
chatSignal.dispatch(currentPanel.id, currentPanel.logs, true);
}
}
}

public function rewriteChatLogs( panelID:uint ):void
{
var currentPanel:ChatPanelVO;
var len:uint = _chatPanels.length;
var i:uint;
for (; i < len; ++i)
{
currentPanel = _chatPanels[i];
if (currentPanel.id == panelID)
break;
}

if (currentPanel)
{
var panelChatLogs:String = '';
var logs:Vector.<ChatVO> = currentPanel.chatlogVO;
var currentChat:ChatVO;
len = logs.length
for (i = 0; i < len; ++i)
{
```

```
currentChat = logs[i];
if (currentChat)
{
panelChatLogs += convertVOToText(currentChat)
}
}
currentPanel.logs = panelChatLogs;
chatSignal.dispatch(currentPanel.id, currentPanel.logs, false);
}
}

public function addOrRemoveBlocked( id:String ):void
{
if (_blockList)
{
var index:int = _blockList.indexOf(id);
if (index != -1)
_blockList.splice(index, 1);
else
_blockList.push(id);
}
}

public function isBlocked( id:String ):Boolean
{
var index:int = -1;
if (_blockList)
index = _blockList.indexOf(id);

return (index == -1) ? false : true;
}

public function isMuted( id:String ):Boolean
{
var found:Boolean;
if (_muteList.indexOf(id) != -1)
found = true;

return found;
}

public function mutePlayer( id:String ):void
{
var index:int = _muteList.indexOf(id);
if (index != -1)
_muteList.splice(index, 1);
else
_muteList.push(id);
}

public
```

```
function overrideDefaultChannelByChannelID( newChannelId:int ):void
{
var newChannel:ChatChannelVO = getChannelFromId(newChannelId);
if (newChannel != null && newChannel.playerSendable == true)
{
defaultChannel = newChannel;

var index:int = DictionaryUtil.getValues(activeChannels).indexOf(defaultChannel);
if (index >= 0)
onDefaultChannelOverriden.dispatch(index);
}
}

private function convertVOToText( vo:ChatVO ):String
{
if (_muteList.indexOf(vo.userID) != -1)
return '';

var message:String;
var localizedMsgId:String;
var tokens:Array = new Array();
var chatDict:Dictionary = new Dictionary();
var locManager:Localization = Localization.instance;

chatDict['[[HexNumber.Color]]'] = vo.channel.channelColor.toString(16);
if (vo.channel.channelID == ChatChannelEnum.SYSTEM)
{
localizedMsgId = _systemMessage;
} else if (vo.channel.channelID == ChatChannelEnum.BROADCAST || vo.channel.channelID ==
ChatChannelEnum.WHISPER)
{
localizedMsgId = _defaultMessage
if (vo.userName == '')
vo.userName = 'Admin';

chatDict['[[String.PlayerName]]'] = vo.userName;
} else if (vo.channel.channelID == ChatChannelEnum.EVENT)
{
localizedMsgId = _defaultMessage
if (vo.userName == '')
vo.userName = 'Unknown';

chatDict['[[String.PlayerName]]'] = vo.userName;
} else
{
localizedMsgId = _message;



var channelName:String = vo.channel.displayName;
chatDict['[[String.NameLink]]']
```

```
= vo.userID + ':' + vo.userName;
chatDict['[[String.PlayerName]]'] = vo.userName;
chatDict['[[FactionHexNumber.Color]]'] =
CommonFunctionUtil.getFactionColor(vo.userFaction).toString(16);

}
chatDict['[[String.Message]]'] = vo.message;

vo.localizedMessage = locManager.localizeStringWithTokens(localizedMsgId, chatDict);

return vo.localizedMessage;
}

private function getChannelNameFromId( channelID:int ):String
{
var channelName:String = '';
switch (channelID)
{
case ChatChannelEnum.GLOBAL:
channelName = _global;
break;
case ChatChannelEnum.FACTION:
channelName = _faction;
break;
case ChatChannelEnum.SECTOR:
channelName = _sector;
break;
case ChatChannelEnum.GROUP:
channelName = _group;
break;
case ChatChannelEnum.ALLIANCE:
channelName = _alliance;
break;
case ChatChannelEnum.MEMBERS:
channelName = _members;
break;
}
return channelName;
}

private function getChannelPlayerSendableFromId( channelID:int ):Boolean
{
var playerSendable:Boolean;
switch (channelID)
{
case ChatChannelEnum.GLOBAL:
case ChatChannelEnum.FACTION:
case ChatChannelEnum.ALLIANCE:
case ChatChannelEnum.MEMBERS:
case ChatChannelEnum.GROUP:
case
```

```
ChatChannelEnum.SECTOR:
playerSendable = true;
break;
case ChatChannelEnum.SYSTEM:
case ChatChannelEnum.BROADCAST:
case ChatChannelEnum.EVENT:
playerSendable = false;
break;
}

return playerSendable;
}

public function get activeChannels():Dictionary { return _activeChannels; }

public function get defaultChannel():ChatChannelVO { return _defaultChannel; }

public function get chatPanels():Vector.<ChatPanelVO> { return _chatPanels; }

public function set blockedList( v:Vector.<String> ):void { _blockList = v; }
public function get blockedList():Vector.<String> { return _blockList; }

public function set defaultChannel( newChannel:ChatChannelVO ):void
{
if (newChannel.channelID in _activeChannels && newChannel.playerSendable == true)
{
_defaultChannel = newChannel;
onDefaultChannelUpdated.dispatch(newChannel);
}
}
}
}

-------------------------------------
File 313: igw\com\model\chat\ChatPanelVO.as
package com.model.chat
{
import flash.utils.Dictionary;

import org.shared.ObjectPool;

public class ChatPanelVO
{
private const MAX_LOG_SIZE:int = 50;

private var _name:String;
private var _id:uint;
private var _logs:Vector.<ChatVO>;
private var _channels:Dictionary;
private var _chatLog:String;
private
```

```
var _defaultSendChannel:ChatChannelVO;

public function ChatPanelVO( name:String, id:uint )
{
_name = name;
_id = id;
_channels = new Dictionary();
_logs = new Vector.<ChatVO>;
_chatLog = '';
}

public function addChatLog( vo:ChatVO, message:String ):void
{
if ( !vo || !message )
return;

_logs.push(vo);
_chatLog += message;
if (_logs.length > MAX_LOG_SIZE)
{
vo = _logs.shift();
var target:String = vo.localizedMessage;
_chatLog = _chatLog.substring(target.length);
ObjectPool.give(vo);
}
}

public function addChannel( channelID:int, channel:ChatChannelVO ):void
{
_channels[channelID] = channel;

if (_defaultSendChannel == null)
_defaultSendChannel = channel;

}

public function removeChannel( channelID:int ):void
{
if (channelID in _channels)
delete _channels[channelID];
}

public function listeningToChannel( channelID:int ):Boolean
{
return (channelID in _channels);
}

public function set logs( v:String ):void { _chatLog = v; }
public function get logs():String { return _chatLog; }
public function get chatlogVO():Vector.<ChatVO> { return _logs; }
public
```

```
function get name():String { return _name; }
public function get id():uint { return _id; }
public function get defaultSendChannel():ChatChannelVO { return _defaultSendChannel; }
}
}

-------------------------------------
File 314: igw\com\model\chat\ChatVO.as
package com.model.chat
{
import flash.utils.getTimer;

public class ChatVO
{
private var _message:String;
private var _timestamp:int;
private var _userID:String;
private var _userFaction:String;
private var _channel:ChatChannelVO;
private var _localizedMessage:String;
private var _userName:String;

public function init( userid:String, username:String, faction:String, channel:ChatChannelVO,
message:String, timestamp:int = -1 ):void
{
_userID = userid;
_userName = username;
_userFaction = faction;
_channel = channel;
_message = message;
_timestamp = (timestamp == -1) ? getTimer() : timestamp;
}

public function get message():String { return _message; }
public function set localizedMessage( v:String ):void { _localizedMessage = v; }
public function get localizedMessage():String { return _localizedMessage; }
public function get timestamp():int { return _timestamp; }
public function get channel():ChatChannelVO { return _channel; }
public function get channelID():int { return _channel.channelID; }
public function get userID():String { return _userID; }
public function get userName():String { return _userName; }
public function set userName( v:String ):void { _userName = v; }
public function get userFaction():String { return _userFaction; }

public function destroy():void
{
_channel = null;
}
}
}

-------------------------------------
```

File 315: igw\com\model\event\EventModel.as

```
package com.model.event
{
import com.model.Model;

import org.osflash.signals.Signal;

public class EventModel extends Model
{
public var onEventsUpdated:Signal;

private var _currentActiveEvent:EventVO;
private var _activeEvents:Vector.<EventVO>;
private var _upcomingEvents:Vector.<EventVO>;

public function EventModel()
{
super();
onEventsUpdated = new Signal(EventVO, Vector.<EventVO>, Vector.<EventVO>);
}

public function addEvents( currentActiveEvent:EventVO, activeEvent:Vector.<EventVO>,
upcomingEvents:Vector.<EventVO>, update:Boolean = false ):void
{
_currentActiveEvent = currentActiveEvent;
_activeEvents = activeEvent;
_upcomingEvents = upcomingEvents;

if (update)
onEventsUpdated.dispatch(_currentActiveEvent, _activeEvents, _upcomingEvents);
}

public function get currentActiveEvent():EventVO { return _currentActiveEvent; }
public function get activeEvents():Vector.<EventVO> { return _activeEvents; }
public function get upcomingEvents():Vector.<EventVO> { return _upcomingEvents; }
}
}
```

File 316: igw\com\model\event\EventVO.as

```
package com.model.event
{
import com.enum.EventStateEnum;
import com.model.prototype.IPrototype;

import flash.utils.getTimer;

public class EventVO
{
private
```

```
var _prototype:IPrototype;

private var _clientTime:int;

private var _state:int;
private var _timeMS:Number;
private var _timeRemainingMS:Number;
private var _timeLeft:Number;

private const _SHOWTIME:Number = 172800000;

public function EventVO( prototype:IPrototype, state:int, timeRemainingMS:Number )
{
_prototype = prototype;
_state = state;
_timeRemainingMS = timeRemainingMS;
_clientTime = getTimer();
}

public function get timeRemainingMS():Number
{
switch (_state)
{
case EventStateEnum.UPCOMING:
case EventStateEnum.RUNNING:
_timeLeft = _timeRemainingMS - (getTimer() - _clientTime);
if (_timeLeft < 0)
_timeLeft = 0;
break;
case EventStateEnum.ENDED:
_timeLeft = 0;
break;
}
return _timeLeft;
}

public function get prototype():IPrototype { return _prototype; }
public function get state():int { return _state; }
public function set state( v:int ):void { _state = v; }
public function get scoreKey():String { return _prototype.getValue('scoreKey'); }
public function get objectiveText():String { return _prototype.getValue('objectiveText'); }
public function get rewardsText():String { return _prototype.getValue('rewardsText'); }
public function get activeEventBuffsText():String { return
_prototype.getValue('activeEventBuffsText'); }
public function get rewards():Array { return _prototype.getValue('rewards'); }
public function get buffsGranted():Array { return _prototype.getValue('buffs'); }
public function get ends():Number { return _prototype.getValue('eventEnds'); }
public function get begins():Number { return _prototype.getValue('eventBegins'); }
public function set timeRemainingMS( v:Number ):void
{
_timeRemainingMS
```

```
= v;
_clientTime = getTimer();
}
public function get isUiTracking():Boolean { return _prototype.getValue('uiTracking'); }
public function get hasScore():Boolean { return _prototype.getValue('hasScore'); }
}
}

------------------------------------
File 317: igw\com\model\fleet\EmptySlotVO.as
package com.model.fleet
{
import com.model.prototype.IPrototype;

public final class EmptySlotVO implements IPrototype
{
public function EmptySlotVO()
{
}

public function getValue( key:String ):*
{
return null;
}

public function getUnsafeValue( key:String ):*
{
return null;
}

public function get asset():String
{
return '';
}

public function get uiAsset():String
{
return '';
}

public function get name():String
{
return '';
}

public function get itemClass():String
{
return '';
}

public
```

```
function get buildTimeSeconds():uint
{
return 0;
}

public function get alloyCost():int
{
return 0;
}

public function get creditsCost():int
{
return 0;
}

public function get energyCost():int
{
return 0;
}

public function get syntheticCost():int
{
return 0;
}
}
}
```

File 318: igw\com\model\fleet\FleetModel.as

```
package com.model.fleet
{
import com.enum.FleetStateEnum;
import com.model.Model;
import com.model.prototype.PrototypeModel;
import com.model.transaction.TransactionVO;
import com.service.server.incoming.data.FleetData;
import com.service.server.incoming.data.ShipData;

import flash.utils.Dictionary;

import org.adobe.utils.DictionaryUtil;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class FleetModel extends Model
{
public var onUpdatedFleetsSignal:Signal;

private var _dirty:Boolean; //set to true when we want to wipe and rebuild all data. should only
be set in rare cases such as a server crash
private
```

```
var _maxAvailableShipSlots:int;
private var _fleets:Vector.<FleetVO>;
private var _lookup:Dictionary;
private var _unassignedShips:Vector.<ShipVO>;
private var _prototypeModel:PrototypeModel;

[PostConstruct]
public function init():void
{
_dirty = false;
_fleets = new Vector.<FleetVO>();
_lookup = new Dictionary();
_unassignedShips = new Vector.<ShipVO>();
onUpdatedFleetsSignal = new Signal(FleetVO);
}


//==========================================================================
//**************************************************************************************************
// FLEETS
//**************************************************************************************************

//==========================================================================

/**
* Take the latest fleet data received from the server and updates an existing fleet or creates a
new one
* @param fleetData The data from the server
*/
public function importFleetData( fleetData:FleetData ):void
{
if (!_lookup[fleetData.id] || _dirty)
{
var fleetVO:FleetVO = ObjectPool.get(FleetVO);
_fleets.push(fleetVO);
_lookup[fleetData.id] = fleetVO;
}

_lookup[fleetData.id].importData(fleetData);
}

/**
* Assigns a ship to a fleet
* @param selectedFleet The target fleet
* @param selectedShip The ship to add
* @param index The index to store the ship at on the fleet
*/
public function assignShipToFleet( selectedFleet:FleetVO, selectedShip:ShipVO, index:int ):void
{
var wasAdded:Boolean = selectedFleet.addShip(selectedShip, index);

if
```

```
(wasAdded)
{
index = _unassignedShips.indexOf(selectedShip);

if (index != -1)
_unassignedShips.splice(index, 1);

updateFleet(selectedFleet);
}
}

public function removeShipFromFleet( id:String, addToUnassignedShips:Boolean = true ):void
{
var ship:ShipVO = _lookup[id];

if (ship && ship.fleetOwner != '' && ship.fleetOwner != null)
{
var fleet:FleetVO = _lookup[ship.fleetOwner];
var wasRemoved:Boolean = fleet.removeShip(ship);

if (wasRemoved)
{
if (addToUnassignedShips)
_unassignedShips.push(ship);

updateFleet(fleet);
}
}
}

public function repairFleet( fleetToRepair:FleetVO, instant:Boolean = false,
transaction:TransactionVO = null ):void
{
if(fleetToRepair == null)
return;

var ship:ShipVO;

if (instant)
{
var ships:Vector.<ShipVO> = fleetToRepair.ships;
var len:uint = ships.length;

for (var i:uint = 0; i < len; ++i)
{
ship = ships[i];

if (ship && ship.currentHealth != ship.maxHealth)
ship.currentHealth = ship.maxHealth;
}

fleetToRepair.state
```

```
= FleetStateEnum.DOCKED;
fleetToRepair.updateFleetStats();
fleetToRepair.currentHealth = 1;
updateFleet(fleetToRepair);
}

else if (transaction && fleetToRepair && _prototypeModel)
{
fleetToRepair.updateFleetStats();
var shipTimeDiff:Number;
var diff:Number = fleetToRepair.repairTime - (transaction.timeRemainingMS) / 1000;
var isNewRepairSystemActive:Boolean =
_prototypeModel.getConstantPrototypeValueByName("isNewRepairSystemActive");

//find a ship to repair
for (i = 0; i < fleetToRepair.ships.length; ++i)
{
ship = fleetToRepair.ships[i];

if (ship && ship.currentHealth != 1)
{
shipTimeDiff = (ship.currentRepairTime <= diff) ? ship.currentRepairTime : diff;
ship.currentHealth += ship.healthGainedASecond * shipTimeDiff;
if (!isNewRepairSystemActive)
{
diff -= shipTimeDiff;

if (diff <= 0)
break;
}
}
}
}
}

public function renameFleet( fleetToRename:FleetVO, newName:String ):void
{
fleetToRename.name = newName;
updateFleet(fleetToRename);
}

public function updateShipFromRefit( id:String, built:Boolean, refiting:Boolean,
refitSuccess:Boolean ):void
{
var currentShip:ShipVO;
var len:uint = _unassignedShips.length;

for (var i:int = 0; i < len; i++)
{
currentShip = _unassignedShips[i];

if
```

```
(currentShip.id == id)
{
currentShip.built = built;
currentShip.refiting = refiting;

if (refitSuccess)
{
var refitMods:Dictionary = currentShip.refitModules;

for (var key:String in refitMods)
currentShip.equipModule(refitMods[key], key);
}
}
}

}

public function getFleetByBattleAddress( address:String ):FleetVO
{
for (var i:int = 0; i < _fleets.length; i++)
{
if (_fleets[i].battleServerAddress == address)
return _fleets[i];
}

return null;
}

public function getFleet( id:String ):FleetVO { return _lookup[id]; }
public function updateFleet( selectedFleet:FleetVO ):void {
onUpdatedFleetsSignal.dispatch(selectedFleet); }


//=============================================================================
//*****************************************************************************
// SHIPS
//*****************************************************************************

//=============================================================================

public function importShipData( shipData:ShipData ):void
{
if (!_lookup[shipData.id] || _dirty)
{
var shipVO:ShipVO = ObjectPool.get(ShipVO);
_lookup[shipData.id] = shipVO;
_lookup[shipData.id].importData(shipData);
//add the ships to the fleet. we only need to do this when the fleet is first created
//subsequent addition or removal of fleets will be taken care of by transactions
var fleetVO:FleetVO = getFleet(shipData.fleetOwner);
if
```

```
(fleetVO)
assignShipToFleet(fleetVO, shipVO, shipData.positionIndex);
else
addShip(shipVO);
}
_lookup[shipData.id].importData(shipData);
}

public function addShip( ship:ShipVO ):void
{
_unassignedShips.push(ship);
_lookup[ship.id] = ship;
}

public function removeShip( id:String ):void
{
var ship:ShipVO = _lookup[id];
if (ship)
{
if (ship.fleetOwner != null && ship.fleetOwner != '')
{
removeShipFromFleet(id, false);

} else
{
var index:int = _unassignedShips.indexOf(ship);
if (index != -1)
_unassignedShips.splice(index, 1);
}
ship.destroy();
delete _lookup[id];
}
}

public function updateShipID( oldID:String, newID:String ):void
{
//TODO fix - crashes sometimes
//ensure that the server didn't beat us to it
if (_lookup[newID] != null)
{
_lookup[oldID] = null;
delete _lookup[oldID];
} else
{
var shipVO:ShipVO = _lookup[oldID];
_lookup[oldID] = null;
delete _lookup[oldID];
if(shipVO != null)
shipVO.id = newID;
_lookup[newID] = shipVO;
}
```

```
}

public function getShip( id:String ):ShipVO { return _lookup[id]; }

public function get builtShipCount():Number
{
var count:uint = 0;

for each (var value:Object in _lookup)
{
if(value as ShipVO)
{
++count;
}
}

return count;
}

public function get canBuildNewShips():Boolean
{
var count:uint = 0;

for each (var value:Object in _lookup)
{
if(value as ShipVO)
{
++count;
}
}

if (count < _maxAvailableShipSlots)
return true;

return false;
}


//=============================================================================
//*****************************************************************************
//*****************************************************************************

//=============================================================================

public function set dirty( v:Boolean ):void { _dirty = v; }
public function set maxAvailableShipSlots( v:Number ):void { _maxAvailableShipSlots = v; }
public function get maxAvailableShipSlots():Number { return _maxAvailableShipSlots; }
public function get fleets():Vector.<FleetVO> { return _fleets; }
public function get ships():Vector.<ShipVO> { return _unassignedShips; }

[Inject]
```

```
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
}
}


------------------------------------
File 319: igw\com\model\fleet\FleetVO.as
package com.model.fleet
{
import com.enum.FleetStateEnum;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.FleetData;
import com.util.statcalc.StatCalcUtil;

public class FleetVO implements IPrototype
{
public var battleServerAddress:String;
public var inBattle:Boolean = false;
public var level:int;
public var ownerID:String;
public var sectorLocationX:int;
public var sectorLocationY:int;
public var currentCargo:int;

private var _repairCreditCost:int = 0;
private var _repairAlloyCost:int = 0;
private var _repairEnergyCost:int = 0;
private var _repairSyntheticCost:int = 0;
private var _repairTime:int = 0;

private var _id:String;
private var _currentHealth:Number;
private var _healthAmount:Number;
private var _powerUsage:Number;
private var _damage:int;
private var _defendTarget:String;
private var _fleetGroupData:Object = {};
private var _groupData:String;
private var _loadSpeed:Number;
private var _maxCargo:int = 0;
private var _maxHealth:int;
private var _name:String;
private var _numOfShips:int;
private var _ships:Vector.<ShipVO>;
private var _starbaseID:String;
private var _state:int;
private var _sector:String;
private var _sectorSpeed:Number = 0;

//For
```

```
new repair system
private var _repairTimeRatingExponent:Number = 0;
private var _repairTimeRatingScale:Number = 0;
private var _repairTimeDamageExponent:Number = 0;
private var _repairResourceRatingExponent:Number = 0;
private var _repairResourceRatingScale:Number = 0;
private var _repairResourceDamageExponent:Number = 0;
private var _repairCreditsRatingExponent:Number = 0;
private var _repairCreditsRatingScale:Number = 0;
private var _repairCreditsDamageExponent:Number = 0;

// For new new repair system
private var _repairTimeDamageAmountShift:Number = 0;
private var _repairTimeDamageScale:Number = 0;
private var _repairTimeDamageFactorShift:Number = 0;
private var _repairResourceDamageAmountShift:Number = 0;
private var _repairResourceDamageScale:Number = 0;
private var _repairResourceDamageFactorShift:Number = 0;
private var _repairCreditsDamageAmountShift:Number = 0;
private var _repairCreditsDamageScale:Number = 0;
private var _repairCreditsDamageFactorShift:Number = 0;

public function FleetVO()
{
_ships = new Vector.<ShipVO>(6, true);

if (PrototypeModel.instance.getConstantPrototypeValueByName("isNewRepairSystemActive"))
{
_repairTimeRatingExponent =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairTimeRatingExponent");
_repairTimeRatingScale =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairTimeRatingScale");
_repairTimeDamageExponent =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairTimeDamageExponent");
_repairResourceRatingExponent =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairResourceRatingExponent");
_repairResourceRatingScale =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairResourceRatingScale");
_repairResourceDamageExponent =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairResourceDamageExponent");
_repairCreditsRatingExponent =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairCreditsRatingExponent");
_repairCreditsRatingScale =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairCreditsRatingScale");
_repairCreditsDamageExponent =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairCreditsDamageExponent");


if
(PrototypeModel.instance.getConstantPrototypeValueByName("useLogarithmicDamageScale"))
{
```

```
_repairTimeDamageAmountShift =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairTimeDamageAmountShift");
_repairTimeDamageScale =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairTimeDamageScale");
_repairTimeDamageFactorShift =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairTimeDamageFactorShift");
_repairResourceDamageAmountShift =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairResourceDamageAmountShift");
_repairResourceDamageScale =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairResourceDamageScale");
_repairResourceDamageFactorShift =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairResourceDamageFactorShift");
_repairCreditsDamageAmountShift =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairCreditsDamageAmountShift");
_repairCreditsDamageScale =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairCreditsDamageScale");
_repairCreditsDamageFactorShift =
PrototypeModel.instance.getConstantPrototypeValueByName("RepairCreditsDamageFactorShift");
}
}
}

public function importData( fleetData:FleetData ):void
{
currentCargo = fleetData.currentCargo;
_defendTarget = fleetData.defendTarget;
_maxCargo = fleetData.cargoCapacity;
_loadSpeed = fleetData.loadSpeed;
currentHealth = fleetData.currentHealth;
_id = fleetData.id;
level = fleetData.level;
_name = fleetData.name;
ownerID = fleetData.ownerID;
sector = fleetData.sector;
sectorLocationX = fleetData.sectorLocationX;
sectorLocationY = fleetData.sectorLocationY;
_starbaseID = fleetData.starbaseID;
}

public function addShip( ship:ShipVO, index:uint ):Boolean
{
if (_ships[index] != ship)
{
_ships[index] = ship;
ship.fleetOwner = id;
updateFleetStats();
return true;
}
return false;
}
```

```
public function removeShip( ship:ShipVO ):Boolean
{
for (var i:int = 0; i < _ships.length; i++)
{
if (_ships[i] == ship)
{
ship.fleetOwner = '';
_ships[i] = null;
updateFleetStats();
return true
}
}
return false;
}

public function updateFleetStats():void
{
_sectorSpeed = 0;
_maxHealth = 1;
_numOfShips = 0;
_currentHealth = 0;
_repairTime = 0;
_damage = 0;
_repairCreditCost = 0;
_repairAlloyCost = 0;
_repairEnergyCost = 0;
_repairSyntheticCost = 0;
_healthAmount = 0;
_maxCargo = 0;
_powerUsage = 0;
var totalWeightedLoadSpeed:Number = 0.0;

var len:uint = _ships.length;
var currentShip:ShipVO;
for (var i:uint = 0; i < len; ++i)
{
currentShip = _ships[i];
if (currentShip)
{
if ((_sectorSpeed > currentShip.mapSpeed || _sectorSpeed == 0) &&
!isNaN(currentShip.mapSpeed))
_sectorSpeed = currentShip.mapSpeed;

_currentHealth += currentShip.currentHealth;
//For New Repair System false
if (!PrototypeModel.instance.getConstantPrototypeValueByName("isNewRepairSystemActive"))
{
_repairCreditCost += currentShip.repairCreditsCost;
_repairAlloyCost
```

```
+= currentShip.repairAlloyCost;
_repairEnergyCost += currentShip.repairEnergyCost;
_repairSyntheticCost += currentShip.repairSyntheticCost;
_repairTime += currentShip.currentRepairTime;
}
_damage += currentShip.shipDps;
_healthAmount += currentShip.healthAmount;
_powerUsage += currentShip.powerUsage;
_maxCargo += currentShip.cargo;
totalWeightedLoadSpeed += currentShip.loadSpeed * currentShip.cargo;

++_numOfShips;
}
}

if (_numOfShips > 0)
_currentHealth /= _numOfShips;
else
_currentHealth = 1;

//For New Repair System true
// Total = RatingScale * ( e ^ ( Rating * RatingExponent ) ) * ( e ^ ( DamagePercent *
DamageExponent ) )
if (PrototypeModel.instance.getConstantPrototypeValueByName("isNewRepairSystemActive"))
{
//For New New Repair System
//Quantity = ( ( Ln( Damage + DamageAmountShift ) * DamageScale ) + DamageFactorShift ) * (
TechScale * Exp( Rating * TechExponent ) )
if
(PrototypeModel.instance.getConstantPrototypeValueByName("useLogarithmicDamageScale"))
{
_repairCreditCost = ( ( Math.log((_maxHealth - _currentHealth) +
_repairCreditsDamageAmountShift)
* _repairCreditsDamageScale) + _repairCreditsDamageFactorShift) *
(_repairCreditsRatingScale * Math.exp(this.level * _repairCreditsRatingExponent));

if(_repairCreditCost < 0)
_repairCreditCost = 0;

_repairAlloyCost = ( ( Math.log((_maxHealth - _currentHealth) +
_repairResourceDamageAmountShift)
* _repairResourceDamageScale) + _repairResourceDamageFactorShift) *
(_repairResourceRatingScale * Math.exp(this.level * _repairResourceRatingExponent));

if(_repairAlloyCost < 0)
_repairAlloyCost = 0;

_repairEnergyCost = ( ( Math.log((_maxHealth - _currentHealth) +
_repairResourceDamageAmountShift)
* _repairResourceDamageScale) + _repairResourceDamageFactorShift) *
(_repairResourceRatingScale
```

```
* Math.exp(this.level * _repairResourceRatingExponent));

if(_repairEnergyCost < 0)
_repairEnergyCost = 0;

_repairSyntheticCost = ( ( Math.log((_maxHealth - _currentHealth) +
_repairResourceDamageAmountShift)
* _repairResourceDamageScale) + _repairResourceDamageFactorShift) *
(_repairResourceRatingScale * Math.exp(this.level * _repairResourceRatingExponent));

if(_repairSyntheticCost < 0)
_repairSyntheticCost = 0;

_repairTime = ( ( Math.log((_maxHealth - _currentHealth) + _repairTimeDamageAmountShift)
* _repairTimeDamageScale) + _repairTimeDamageFactorShift) * (_repairTimeRatingScale *
Math.exp(this.level * _repairTimeRatingExponent));

if(_repairTime < 0)
_repairTime = 0;
}
else
{
_repairCreditCost = _repairCreditsRatingScale * (Math.exp(this.level *
_repairCreditsRatingExponent)) * (Math.exp((_maxHealth - _currentHealth) *
_repairCreditsDamageExponent));
_repairAlloyCost = _repairResourceRatingScale * (Math.exp(this.level *
_repairResourceRatingExponent)) * (Math.exp((_maxHealth - _currentHealth) *
_repairResourceDamageExponent));
_repairEnergyCost = _repairResourceRatingScale * (Math.exp(this.level *
_repairResourceRatingExponent)) * (Math.exp((_maxHealth - _currentHealth) *
_repairResourceDamageExponent));
_repairSyntheticCost = _repairResourceRatingScale * (Math.exp(this.level *
_repairResourceRatingExponent)) * (Math.exp((_maxHealth - _currentHealth) *
_repairResourceDamageExponent));
_repairTime = _repairTimeRatingScale * (Math.exp(this.level * _repairTimeRatingExponent)) *
(Math.exp((_maxHealth - _currentHealth) * _repairTimeDamageExponent));
}
}

// TODO - we need to update this if the buff expires
_repairTime = StatCalcUtil.baseStatCalc("repairSpeed", _repairTime);
}

public function GetFleetHealthFromRepairTimeRemaining():Number
{
// / time \
// ln | ------------ | - ratingExponent * rating
// damage = \ ratingScale /
// ------------------------------------------------
// damageExponent

//Take
```

```
care of floating point strangeness by multiplying value by 100 in the round then dividing by 100
to get 2 decimals
if (PrototypeModel.instance.getConstantPrototypeValueByName("isNewRepairSystemActive"))
{
if
(PrototypeModel.instance.getConstantPrototypeValueByName("useLogarithmicDamageScale"))
{
return 1 - ( Math.exp( ( ( ( _repairTime ) * Math.exp( -this.level * _repairTimeRatingExponent ) ) /
( _repairTimeDamageScale
* _repairTimeRatingScale ) ) - ( _repairTimeDamageFactorShift / _repairTimeDamageScale ) ) -
_repairTimeDamageAmountShift );
}
else
return Math.round(100 * (1 - ( ( Math.log( (_repairTime) /
_repairTimeRatingScale ) - (_repairTimeRatingExponent * this.level) ) /
_repairTimeDamageExponent ))) / 100;
}
else
return (_maxHealth - _currentHealth);


}

public function getShipIndexByID( id:String ):int
{
for (var i:int = 0; i < _ships.length; i++)
{
if (_ships[i] && _ships[i].id == id)
return i;
}
return -1;
}

public function getShipIDByIndex( index:int ):String
{
if (index < _ships.length && _ships[index] != null)
return _ships[index].id;
return null;
}

public function getValue( key:String ):* { return null; }
public function getUnsafeValue( key:String ):* { return null; }

public function get asset():String
{
var fleetType:String = '';
var len:uint = _ships.length;
for (var i:uint = 0; i < len; ++i)
{
if (_ships[i] != null)
{
```

```
fleetType = _ships[i].prototypeVO.asset;
break;
}
}

return fleetType;
}

public function get cargoPercent():Number { return _maxCargo ? int(currentCargo / _maxCargo
* 100) : 0; }
public function get currentHealth():Number { return _currentHealth; }
public function set currentHealth( curHealth:Number ):void { _currentHealth = curHealth; }
public function get healthAmount():Number { return _healthAmount; }
public function get damage():int { return _damage; }
public function get defendTarget():String { return _defendTarget; }
public function set defendTarget( v:String ):void { _defendTarget = v; }
public function get fleetGroupData():Object { return _fleetGroupData; }
public function get itemClass():String { return ''; }
public function get id():String { return _id; }
public function get loadSpeed():Number { return _loadSpeed; }
public function get maxCargo():int { return _maxCargo; }
public function get maxHealth():int { return _maxHealth; }
public function get powerUsage():Number { return _powerUsage; }
public function get name():String { return _name; }
public function set name( v:String ):void { _name = v; }
public function get needsRepair():Boolean { return (_numOfShips > 0) ? _currentHealth < 1.0 :
false; }
public function get numOfShips():int { return _numOfShips; }
public function get repairTime():int { return _repairTime; }
public function get ships():Vector.<ShipVO> { return _ships; }
public function get state():int { return _state; }
public function set state( state:int ):void { _state = state; }
public function get uiAsset():String { return ''; }
public function get sectorSpeed():Number { return _sectorSpeed; }
public function get starbaseID():String { return _starbaseID; }

public function get sector():String { return _sector; }
public function set sector( sector:String ):void
{
_sector = sector;
if (_sector == '' || _sector == null)
state = FleetStateEnum.DOCKED;
}

public function get alloyCost():int { return _repairAlloyCost; }
public function get creditsCost():int { return _repairCreditCost; }
public function get energyCost():int { return _repairEnergyCost; }
public function get syntheticCost():int { return _repairSyntheticCost; }

public
```

```
function get buildTimeSeconds():uint { return _repairTime; }

public function destroy():void
{
}
}
}

-----------------------------------
File 320: igw\com\model\fleet\ShipVO.as
package com.model.fleet
{
import com.enum.TypeEnum;
import com.enum.server.StarbaseBuildStateEnum;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.ShipData;
import com.util.CommonFunctionUtil;
import com.util.statcalc.StatCalcUtil;

import flash.utils.Dictionary;

import org.adobe.utils.StringUtil;

public class ShipVO implements IPrototype
{
public static const EMPTY_SLOT:IPrototype = new EmptySlotVO();

public var id:String;
public var ownerID:String;
public var fleetOwner:String;
public var built:Boolean;
public var refiting:Boolean;
public var modules:Dictionary = new Dictionary();
public var refitModules:Dictionary = new Dictionary();
public var inFleet:Boolean = false;
public var shipName:String;
public var refitShipName:String;

private var _buildCreditCost:int = 0;
private var _buildAlloyCost:int = 0;
private var _buildEnergyCost:int = 0;
private var _buildSyntheticCost:int = 0;
private var _buildTime:int = 0;

private var _repairCreditCost:int = 0;
private var _repairAlloyCost:int = 0;
private var _repairEnergyCost:int = 0;
private var _repairSyntheticCost:int = 0;
private var _repairTime:Number = 0;

private
```

```
var _scrapCreditCost:int = 0;
private var _scrapAlloyCost:int = 0;
private var _scrapEnergyCost:int = 0;
private var _scrapSyntheticCost:int = 0;

public var powerUsage:int = 0;

private var _currentHealth:Number = 1;
private var _healthAmount:Number;
private var _shipDps:int;
private var _maxWeaponRange:int;
private var _rating:int;
private var _prototypeVO:IPrototype;
private var _rank:int;
private var _tooltip:String;
private var _healthPercentGainASecond:Number;

public function importData( shipData:ShipData ):void
{
built = (shipData.buildState == StarbaseBuildStateEnum.DONE) ? true : false;
_currentHealth = shipData.currentHealth;
fleetOwner = shipData.fleetOwner;
id = shipData.id;
shipName = shipData.shipName;
refitShipName = shipName;
modules = shipData.modules;
ownerID = shipData.ownerID;
_prototypeVO = shipData.prototype;
refitModules = shipData.refitModules ? shipData.refitModules : new Dictionary();
calculateCosts();
}

public function clone():ShipVO
{
var vo:ShipVO = new ShipVO();
vo.prototypeVO = _prototypeVO;
vo.ownerID = ownerID;
vo.fleetOwner = fleetOwner;
vo.built = built;
vo.refiting = refiting;

vo.shipName = shipName;
vo.refitShipName = refitShipName;

var key:String;

for (key in modules)
{
vo.equipModule(modules[key], key);
}

for
```

```
(key in refitModules)
{
vo.equipRefitModule(refitModules[key], key);
}

return vo;
}

public function setNewShipName( name:String ):void
{
refitShipName = name;
calculateCosts();
}

public function equipModule( vo:IPrototype, index:String ):void
{
modules[index] = vo;
calculateCosts();
}

public function equipRefitModule( vo:IPrototype, index:String ):void
{
if (vo != null && modules[index] != null && modules[index].name == vo.name)
delete refitModules[index];
else
refitModules[index] = (vo == null) ? EMPTY_SLOT : vo;
calculateCosts();
}

public function getUnsafeValue( key:String ):* { return prototypeVO.getUnsafeValue(key); }
public function getValue( key:String ):* { return prototypeVO.getValue(key); }

/** This is actually the PERCENTAGE of current health, expressed as a floating pt number
between 0 and 1 */
public function get currentHealth():Number { return _currentHealth; }
public function set currentHealth( currentHealth:Number ):void
{
if (currentHealth > maxHealth)
_currentHealth = maxHealth;
else
_currentHealth = currentHealth;
}

public function get prototypeVO():IPrototype { return _prototypeVO; }
public function set prototypeVO( prototypeVO:IPrototype ):void
{
//just received a new prototype.
if (_prototypeVO && _prototypeVO != prototypeVO)
{
modules = new Dictionary();
refitModules
```

```
= new Dictionary();
}
_prototypeVO = prototypeVO;
calculateCosts();
}

public function calculateCosts():void
{
var refitVO:IPrototype;
var vo:IPrototype;
var key:String;

_buildCreditCost = 0;
_buildAlloyCost = 0;
_buildEnergyCost = 0;
_buildSyntheticCost = 0;
_scrapCreditCost = 0;
_scrapAlloyCost = 0;
_scrapEnergyCost = 0;
_scrapSyntheticCost = 0;
_buildTime = 0;
powerUsage = 0;
_shipDps = 0;
var currentRange:int = 0;
var id:Number = _prototypeVO.getValue('id');
_healthAmount = _prototypeVO.getValue('health');

//repair cost of the hull
_repairCreditCost = _prototypeVO.getValue('repairCredits');
_repairAlloyCost = _prototypeVO.getValue('repairAlloy');
_repairEnergyCost = _prototypeVO.getValue('repairEnergy');
_repairSyntheticCost = _prototypeVO.getValue('repairSynthetic');
_repairTime = _prototypeVO.getValue('repairTimeSeconds');

//build cost of the hull
if (!built)
{
_buildCreditCost = _prototypeVO.creditsCost;
_buildAlloyCost = _prototypeVO.alloyCost;
_buildEnergyCost = _prototypeVO.energyCost;
_buildSyntheticCost = _prototypeVO.syntheticCost;
_buildTime = _prototypeVO.buildTimeSeconds;
} else
{
_scrapCreditCost += _prototypeVO.creditsCost;
_scrapAlloyCost += _prototypeVO.alloyCost;
_scrapEnergyCost += _prototypeVO.energyCost;
_scrapSyntheticCost += _prototypeVO.syntheticCost;
}

for
```

```
(key in modules)
{
vo = modules[key];
refitVO = refitModules[key];
//repair cost of components
if (vo != null)
{
_repairCreditCost += vo.getValue('repairCredits');
_repairAlloyCost += vo.getValue('repairAlloy');
_repairEnergyCost += vo.getValue('repairEnergy');
_repairSyntheticCost += vo.getValue('repairSynthetic');
_repairTime += vo.getValue('repairTimeSeconds');
}

vo = (refitVO != null) ? (refitVO != EMPTY_SLOT) ? refitVO : null : vo;
if (vo)
{
if (vo == refitVO || !built)
{
_buildCreditCost += vo.creditsCost;
_buildAlloyCost += vo.alloyCost;
_buildEnergyCost += vo.energyCost;
_buildSyntheticCost += vo.syntheticCost;
_buildTime += vo.buildTimeSeconds;
}

if (built)
{
_scrapCreditCost += vo.creditsCost;
_scrapAlloyCost += vo.alloyCost;
_scrapEnergyCost += vo.energyCost;
_scrapSyntheticCost += vo.syntheticCost;
}

powerUsage += vo.getValue("powerCost");
_shipDps += calcSingleDps(vo, key);
var calcedHealth:Number = calcHealth(vo);
if (calcedHealth != 0)
_healthAmount = calcedHealth;
currentRange = calcMaxWeaponRange(vo, key);
if (currentRange > _maxWeaponRange)
_maxWeaponRange = currentRange;
}
}

if(shipName != refitShipName)
{
_buildCreditCost += 50000;
_buildTime += 5;
}

_scrapCreditCost
```

```
*= 0.2;
_scrapAlloyCost *= 0.2;
_scrapEnergyCost *= 0.2;
_scrapSyntheticCost *= 0.2;

_repairCreditCost *= PrototypeModel.SHIP_REPAIR_RESOURCE_COST_SCALAR;
_repairAlloyCost *= PrototypeModel.SHIP_REPAIR_RESOURCE_COST_SCALAR;
_repairEnergyCost *= PrototypeModel.SHIP_REPAIR_RESOURCE_COST_SCALAR;
_repairSyntheticCost *= PrototypeModel.SHIP_REPAIR_RESOURCE_COST_SCALAR;
_repairTime *= PrototypeModel.SHIP_REPAIR_TIME_SCALAR;

_buildCreditCost *= PrototypeModel.SHIP_BUILD_RESOURCE_COST_SCALAR;
_buildAlloyCost *= PrototypeModel.SHIP_BUILD_RESOURCE_COST_SCALAR;
_buildEnergyCost *= PrototypeModel.SHIP_BUILD_RESOURCE_COST_SCALAR;
_buildSyntheticCost *= PrototypeModel.SHIP_BUILD_RESOURCE_COST_SCALAR;
_buildTime *= PrototypeModel.SHIP_BUILD_TIME_SCALAR;

_healthPercentGainASecond = (_healthAmount / _repairTime) / _healthAmount;
}

public function get rank():int { return _rank; }
public function get healthAmount():Number { return _healthAmount; }
public function get rotationSpeed():Number { return _prototypeVO.getValue('rotationSpeed'); }
public function get mapSpeed():Number { return _prototypeVO.getValue('mapSpeed'); }
public function get cargo():int { return _prototypeVO.getValue('cargo'); }
public function get maxSpeed():Number { return _prototypeVO.getValue('maxSpeed'); }
public function get loadSpeed():Number { return _prototypeVO.getValue('loadSpeed'); }
public function get shipDps():int { return _shipDps; }
public function get rating():int { return _rating; }
public function get power():int { return _prototypeVO.getValue('power'); }
public function get maxHealth():int { return 1; }
public function get evasion():Number { return _prototypeVO.getValue('evasion'); }
public function get armor():Number { return _prototypeVO.getValue('armor'); }
public function get profile():Number { return _prototypeVO.getValue('profile'); }
public function get masking():Number { return _prototypeVO.getValue('masking'); }
public function get maxRange():Number { return _maxWeaponRange; }
public function get rarity():Number { return _prototypeVO.getValue('rarity'); }
public function get slots():Array { return _prototypeVO.getValue('slots'); }
public function get health():Number { return _prototypeVO.getValue('health'); }

public function get asset():String { return prototypeVO.asset; }
public function get uiAsset():String { return prototypeVO.uiAsset; }

public function get name():String { return prototypeVO.name; }
public function get itemClass():String { return prototypeVO.itemClass; }
public function get buildTimeSeconds():uint { return _buildTime; }

public function get alloyCost():int { return _buildAlloyCost; }
public function get creditsCost():int { return _buildCreditCost; }
public function get energyCost():int { return _buildEnergyCost; }
public
```

```
function get syntheticCost():int { return _buildSyntheticCost; }

public function get repairAlloyCost():int { return (1 - _currentHealth) * _repairAlloyCost; }
public function get repairCreditsCost():int { return (1 - _currentHealth) * _repairCreditCost; }
public function get repairEnergyCost():int { return (1 - _currentHealth) * _repairEnergyCost; }
public function get repairSyntheticCost():int { return (1 - _currentHealth) * _repairSyntheticCost; }

public function get scrapAlloyCost():int { return _scrapAlloyCost; }
public function get scrapCreditsCost():int { return _scrapCreditCost; }
public function get scrapEnergyCost():int { return _scrapEnergyCost; }
public function get scrapSyntheticCost():int { return _scrapSyntheticCost; }

public function get healthGainedASecond():Number { return _healthPercentGainASecond; }

public function get currentRepairTime():Number { return (1 - _currentHealth) * _repairTime; }

public function get accelerationTime():Number
{
var accelTime:Number = StatCalcUtil.entityStatCalc(this, 'accelerationTime');
var maxSpeed:Number = StatCalcUtil.entityStatCalc(this, 'maxSpeed');
return maxSpeed / accelTime;
}

private function calcSingleDps( mod:IPrototype, slot:String ):Number
{
var slotType:String = mod.getValue('slotType');
if (!(slotType == "Weapon" || slotType == "Spinal" || slotType == "Arc" || slotType == "Drone" ||
slotType == "BaseTurret"))
return 0;

var fireTime:Number = StatCalcUtil.entityStatCalc(this, 'fireTime', 0, mod, slot);
var burstSize:Number = StatCalcUtil.entityStatCalc(this, 'burstSize', 0, mod, slot);
var reloadTime:Number = StatCalcUtil.entityStatCalc(this, 'reloadTime', 0, mod, slot);
var chargeTime:Number = StatCalcUtil.entityStatCalc(this, 'chargeTime', 0, mod, slot);
var volleySize:Number = StatCalcUtil.entityStatCalc(this, 'volleySize', 0, mod, slot);
var duration:Number = StatCalcUtil.entityStatCalc(this, 'duration', 0, mod, slot);
var tickRate:Number = StatCalcUtil.entityStatCalc(this, 'tickRate', 0, mod, slot);
var damageTime:Number = StatCalcUtil.entityStatCalc(this, 'damageTime', 0, mod, slot);
var maxDrones:Number = StatCalcUtil.entityStatCalc(this, 'maxDrones', 0, mod, slot);
var damage:Number = StatCalcUtil.entityStatCalc(this, 'damage', 0, mod, slot);

// Do the appropriate calculation based on the attack type
var type:int = mod.getValue('attackMethod');
var totalDamage:Number = 0;
var totalPeriod:Number = 0;
if (type == 1 || type == 2 || type == 3)
{
// Beams and Projectiles
totalDamage = damage * Math.max(burstSize, 1) * Math.max(volleySize, 1);
totalPeriod = (fireTime * burstSize) + reloadTime + chargeTime;
}
```

```
else if (type == 4)
{
// Areas
totalDamage = damage * Math.max(burstSize, 1) * Math.max(volleySize, 1) * Math.max(duration
/ Math.max(tickRate, 1), 1);
totalPeriod = (fireTime * burstSize) + reloadTime + chargeTime + duration;
} else if (type == 5)
{
// Drones
totalDamage = damage * Math.max(maxDrones, 1);
totalPeriod = damageTime;
}

var DPS:Number = totalDamage / totalPeriod;
return DPS;
}

private function calcMaxWeaponRange( mod:IPrototype, slot:String ):Number
{
var range:Number = 0;

if (!mod || mod.getValue('slotType') == 'Defense' || mod.getValue('slotType') == 'Tech' ||
mod.getValue('slotType') == 'Structure')
range = 0;
else
range = StatCalcUtil.entityStatCalc(this, 'maxRange', 0, mod, slot);

return range;
}

private function calcHealth( mod:IPrototype ):Number
{
return StatCalcUtil.entityStatCalc(this, 'health');
}

public function get tooltip():String
{
_tooltip = StringUtil.getTooltip(String(TypeEnum.SHIP_BUILT_TT), this);
return _tooltip;
}

public function destroy():void
{
}
}
}
```

File

321: igw\com\model\leaderboards\LeaderboardEntryVO.as

```
package com.model.leaderboards
{
import com.service.server.incoming.data.LeaderboardAllianceData;
import com.service.server.incoming.data.LeaderboardPlayerData;

public class LeaderboardEntryVO
{
private var _isAlliance:Boolean;

private var _key:String;
private var _name:String;
private var _racePrototype:String;

private var _sectorOwner:String;
private var _allianceKey:String;
private var _allianceName:String;

private var _experience:uint;
private var _experienceRank:int;

private var _commendationPointsPvE:uint;
private var _commendationPointsPvP:uint;
private var _commendiationCombinedRank:int;
private var _totalCommendationScore:uint;

private var _KTDR:Number;
private var _kdrRank:int;

private var _baseRating:Number;
private var _baseRatingRank:int;

private var _wins:uint;
private var _winsRank:int;

private var _losses:uint;

private var _highestFleetRating:int;
private var _highestFleetRank:int;

private var _avgHighestFleetRating:int;
private var _avgHighestFleetRank:int;

private var _highestBaseRating:int;
private var _highestBaseRanking:int;

private var _blueprintPartsCollected:int;
private var _blueprintPartsRank:int;

private var _numOfMembers:int;
private
```

```
var _numOfMembersRank:int;


private var _qualifiedWinsPvP:Number;
private var _BubbleHoursGranted:Number;
private var _currentPVPEvent:Number;
private var _currentPVPEventQuarter:Number;
private var _CreditsTradeRoute:Number;
private var _ResourcesTradeRoute:Number;
private var _ResourcesSalvaged:Number;
private var _CreditsBounty:Number;
private var _WinsVsBase:Number;


private var _qualifiedWinsPvPRank:int;
private var _BubbleHoursGrantedRank:int;
private var _currentPVPEventRank:int;
private var _currentPVPEventQuarterRank:int;
private var _CreditsTradeRouteRank:int;
private var _ResourcesTradeRouteRank:int;
private var _ResourcesSalvagedRank:int;
private var _CreditsBountyRank:int;
private var _WinsVsBaseRank:int;

public function setUpFromPlayerData( leaderboardPlayerData:LeaderboardPlayerData ):void
{
_key = leaderboardPlayerData.playerKey;
_name = leaderboardPlayerData.name;
_racePrototype = leaderboardPlayerData.racePrototype;
_sectorOwner = leaderboardPlayerData.sectorOwner;
_allianceKey = leaderboardPlayerData.allianceKey;
_allianceName = leaderboardPlayerData.allianceName;

_experience = leaderboardPlayerData.experience;
_experienceRank = leaderboardPlayerData.experienceRank;

_commendationPointsPvE = leaderboardPlayerData.commendationPointsPvE;
_commendationPointsPvP = leaderboardPlayerData.commendationPointsPvP;
_commendiationCombinedRank = leaderboardPlayerData.commendiationCombinedRank;

_baseRating = leaderboardPlayerData.baseRating;
_baseRatingRank = leaderboardPlayerData.baseRatingRank;

_wins = leaderboardPlayerData.wins;
_winsRank = leaderboardPlayerData.winsRank;

_losses = leaderboardPlayerData.losses;
_kdrRank = leaderboardPlayerData.kdrRank;

_highestFleetRating = leaderboardPlayerData.highestFleetRating;
_highestFleetRank
```

```
= leaderboardPlayerData.highestFleetRank;

_blueprintPartsCollected = leaderboardPlayerData.blueprintPartsCollected;
_blueprintPartsRank = leaderboardPlayerData.blueprintPartsRank;


_qualifiedWinsPvP = leaderboardPlayerData.qualifiedWinsPvP;
_BubbleHoursGranted = leaderboardPlayerData.BubbleHoursGranted;
_currentPVPEvent = leaderboardPlayerData.currentPVPEvent;
_currentPVPEventQuarter = leaderboardPlayerData.currentPVPEventQuarter;
_CreditsTradeRoute = leaderboardPlayerData.CreditsTradeRoute;
_ResourcesTradeRoute = leaderboardPlayerData.ResourcesTradeRoute;
_ResourcesSalvaged = leaderboardPlayerData.ResourcesSalvaged;
_CreditsBounty = leaderboardPlayerData.CreditsBounty;
_WinsVsBase = leaderboardPlayerData.WinsVsBase;

_qualifiedWinsPvPRank = leaderboardPlayerData.qualifiedWinsPvPRank;
_BubbleHoursGrantedRank = leaderboardPlayerData.BubbleHoursGrantedRank;
_currentPVPEventRank = leaderboardPlayerData.currentPVPEventRank;
_currentPVPEventQuarterRank = leaderboardPlayerData.currentPVPEventQuarterRank;
_CreditsTradeRouteRank = leaderboardPlayerData.CreditsTradeRouteRank;
_ResourcesTradeRouteRank = leaderboardPlayerData.ResourcesTradeRouteRank;
_ResourcesSalvagedRank = leaderboardPlayerData.ResourcesSalvagedRank;
_CreditsBountyRank = leaderboardPlayerData.CreditsBountyRank;
_WinsVsBaseRank = leaderboardPlayerData.WinsVsBaseRank;


if (_losses != 0)
_KTDR = Math.round(_wins * 100 / _losses) / 100;
else
_KTDR = _wins;
}

public function setUpFromAllianceData( leaderboardAllianceData:LeaderboardAllianceData
):void
{
_isAlliance = true;

_key = leaderboardAllianceData.key;
_name = leaderboardAllianceData.name;
_racePrototype = leaderboardAllianceData.factionPrototype;

_experience = leaderboardAllianceData.totalExperience;
_experienceRank = leaderboardAllianceData.totalExperienceRank;

_commendationPointsPvE = leaderboardAllianceData.totalCommendationPointsPvE;
_commendationPointsPvP = leaderboardAllianceData.totalCommendationPointsPvP;
_commendiationCombinedRank = leaderboardAllianceData.totalCommendiationCombinedRank;
_totalCommendationScore = _commendationPointsPvE + _commendationPointsPvP;

_baseRating
```

```
= leaderboardAllianceData.avgRating;
_baseRatingRank = leaderboardAllianceData.avgBaseRatingRank;

_highestBaseRating = leaderboardAllianceData.totalRating;
_highestBaseRanking = leaderboardAllianceData.totalBaseRatingRank

_wins = leaderboardAllianceData.totalWins;
_winsRank = leaderboardAllianceData.totalWinsRank;

_losses = leaderboardAllianceData.totalLosses;
_kdrRank = leaderboardAllianceData.totalKdrRank;

_avgHighestFleetRank = leaderboardAllianceData.avgHighestFleetRank;
_avgHighestFleetRating = leaderboardAllianceData.avgHighestFleetRating;

_highestFleetRating = leaderboardAllianceData.totalHighestFleetRating;
_highestFleetRank = leaderboardAllianceData.totalHighestFleetRank;

_blueprintPartsCollected = leaderboardAllianceData.totalBlueprintPartsCollected;
_blueprintPartsRank = leaderboardAllianceData.totalBlueprintPartsRank;

_numOfMembers = leaderboardAllianceData.numMembers;
_numOfMembersRank = leaderboardAllianceData.numMembersRank;

_qualifiedWinsPvP = leaderboardAllianceData.totalqualifiedWinsPvP;
_BubbleHoursGranted = leaderboardAllianceData.totalBubbleHoursGranted;
_currentPVPEvent = leaderboardAllianceData.totalcurrentPVPEvent;
_currentPVPEventQuarter = leaderboardAllianceData.totalcurrentPVPEventQuarter;
_CreditsTradeRoute = leaderboardAllianceData.totalCreditsTradeRoute;
_ResourcesTradeRoute = leaderboardAllianceData.totalResourcesTradeRoute;
_ResourcesSalvaged = leaderboardAllianceData.totalResourcesSalvaged;
_CreditsBounty = leaderboardAllianceData.totalCreditsBounty;
_WinsVsBase = leaderboardAllianceData.totalWinsVsBase;

_qualifiedWinsPvPRank = leaderboardAllianceData.totalqualifiedWinsPvPRank;
_BubbleHoursGrantedRank = leaderboardAllianceData.totalBubbleHoursGrantedRank;
_currentPVPEventRank = leaderboardAllianceData.totalcurrentPVPEventRank;
_currentPVPEventQuarterRank = leaderboardAllianceData.totalcurrentPVPEventQuarterRank;
_CreditsTradeRouteRank = leaderboardAllianceData.totalCreditsTradeRouteRank;
_ResourcesTradeRouteRank = leaderboardAllianceData.totalResourcesTradeRouteRank;
_ResourcesSalvagedRank = leaderboardAllianceData.totalResourcesSalvagedRank;
_CreditsBountyRank = leaderboardAllianceData.totalCreditsBountyRank;
_WinsVsBaseRank = leaderboardAllianceData.totalWinsVsBaseRank;

if (_losses != 0)
_KTDR = Math.round(_wins * 100 / _losses) / 100;
else
_KTDR = _wins;
}


public
```

```
function get isAlliance():Boolean { return _isAlliance; }
public function get key():String { return _key; }
public function get name():String { return _name; }
public function get racePrototype():String { return _racePrototype; }
public function get sectorOwner():String { return _sectorOwner; }
public function get allianceKey():String { return _allianceKey; }
public function get allianceName():String { return _allianceName; }

public function get commendationPointsPvE():uint { return _commendationPointsPvE; }
public function get commendationPointsPvP():uint { return _commendationPointsPvP; }
public function get losses():Number { return _losses; }

public function get wins():Number { return _wins; }
public function get winsRank():Number { return _winsRank; }

public function get baseRating():Number { return _baseRating; }
public function get baseRatingRank():Number { return _baseRatingRank; }

public function get highestBaseRating():Number { return _highestBaseRating; }
public function get highestBaseRanking():Number { return _highestBaseRanking; }

public function get experience():uint { return _experience; }
public function get experienceRank():Number { return _experienceRank; }

public function get commendationScore():uint { return _commendationPointsPvE +
_commendationPointsPvP; }
public function get commendiationCombinedRank():Number { return
_commendiationCombinedRank; }

public function get totalCommendationScore():uint { return _totalCommendationScore; }

public function get ktdRatio():Number { return _KTDR; }
public function get kdrRank():Number { return _kdrRank; }

public function get highestFleetRating():Number { return _highestFleetRating; }
public function get highestFleetRank():Number { return _highestFleetRank; }

public function get avgHighestFleetRank():int { return _avgHighestFleetRank; }
public function get avgHighestFleetRating():int { return _avgHighestFleetRating; }

public function get blueprintPartsCollected():Number { return _blueprintPartsCollected; }
public function get blueprintPartsRank():Number { return _blueprintPartsRank; }



public function get numOfMembers():Number { return _numOfMembers; }
public function get numOfMembersRank():Number { return _numOfMembersRank; }


public function get qualifiedWinsPvP():Number { return _qualifiedWinsPvP; }
public
```

```
function get BubbleHoursGranted():Number { return _BubbleHoursGranted; }
public function get currentPVPEvent():Number { return _currentPVPEvent; }
public function get currentPVPEventQuarter():Number { return _currentPVPEventQuarter; }
public function get CreditsTradeRoute():Number { return _CreditsTradeRoute; }
public function get ResourcesTradeRoute():Number { return _ResourcesTradeRoute; }
public function get ResourcesSalvaged():Number { return _ResourcesSalvaged; }
public function get CreditsBounty():Number { return _CreditsBounty; }
public function get WinsVsBase():Number { return _WinsVsBase; }



public function get qualifiedWinsPvPRank():int { return _qualifiedWinsPvPRank; }
public function get BubbleHoursGrantedRank():int { return _BubbleHoursGrantedRank; }
public function get currentPVPEventRank():int { return _currentPVPEventRank; }
public function get currentPVPEventQuarterRank():int { return _currentPVPEventQuarterRank; }
public function get CreditsTradeRouteRank():int { return _CreditsTradeRouteRank; }
public function get ResourcesTradeRouteRank():int { return _ResourcesTradeRouteRank; }
public function get ResourcesSalvagedRank():int { return _ResourcesSalvagedRank; }
public function get CreditsBountyRank():int { return _CreditsBountyRank; }
public function get WinsVsBaseRank():int { return _WinsVsBaseRank; }
}
}

------------------------------------
File 322: igw\com\model\leaderboards\LeaderboardModel.as
package com.model.leaderboards
{
import com.enum.LeaderboardEnum;
import com.model.Model;
import com.service.server.incoming.data.LeaderboardData;
import com.service.server.incoming.leaderboard.LeaderboardResponse;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;

public class LeaderboardModel extends Model
{
private var _leaderboardEntries:Dictionary;
private var _lastRequestedType:int;
private var _lastRequestedScope:int;
public var onLeaderboardDataUpdated:Signal;

public function LeaderboardModel()
{
_leaderboardEntries = new Dictionary();
onLeaderboardDataUpdated = new Signal(LeaderboardVO);
_lastRequestedScope = LeaderboardEnum.PLAYER_GLOBAL;
_lastRequestedType = LeaderboardEnum.BASE_RATING;

}
```

```
public function updateLeaderboardEntry( response:LeaderboardResponse ):void
{
var currentEntry:LeaderboardVO;
var leaderboardData:LeaderboardData = response.leaderboardData;
_lastRequestedType = leaderboardData.leaderboardType;
_lastRequestedScope = leaderboardData.leaderboardScope;
if (_lastRequestedScope in _leaderboardEntries && _lastRequestedType in
_leaderboardEntries[_lastRequestedScope])
currentEntry = _leaderboardEntries[_lastRequestedScope][_lastRequestedType];
else
{
currentEntry = new LeaderboardVO();
if (_leaderboardEntries[_lastRequestedScope] == null)
_leaderboardEntries[_lastRequestedScope] = new Dictionary;
}

currentEntry.update(leaderboardData);

_leaderboardEntries[_lastRequestedScope][_lastRequestedType] = currentEntry;

onLeaderboardDataUpdated.dispatch(currentEntry)
}

public function getLeaderboardDataByType( type:int, scope:int ):LeaderboardVO
{
_lastRequestedType = type;
_lastRequestedScope = scope;
return getLeaderboardData;
}

public function get getLeaderboardData():LeaderboardVO
{
var currentEntry:LeaderboardVO
if (_lastRequestedScope in _leaderboardEntries && _lastRequestedType in
_leaderboardEntries[_lastRequestedScope])
{
currentEntry = _leaderboardEntries[_lastRequestedScope][_lastRequestedType];
if (currentEntry.needsUpdate())
return null;
}
return currentEntry;
}

public function get lastRequestedType():int
{
return _lastRequestedType;
}

public
```

```
function set lastRequestedType( v:int ):void
{
_lastRequestedType = v;
}

public function get lastLeaderboardScope():int
{
return _lastRequestedScope;
}

public function set lastLeaderboardScope( v:int ):void
{
_lastRequestedScope = v;
}
}
}

------------------------------------
File 323: igw\com\model\leaderboards\LeaderboardVO.as
package com.model.leaderboards
{
import com.service.server.incoming.data.LeaderboardAllianceData;
import com.service.server.incoming.data.LeaderboardData;
import com.service.server.incoming.data.LeaderboardPlayerData;

import flash.utils.getTimer;

public class LeaderboardVO
{
private var _type:int;
private var _players:Vector.<LeaderboardEntryVO>;
private var _alliances:Vector.<LeaderboardEntryVO>;
private var _lastUpdate:Number;

private var _refreshTime:Number = 300000;

public function LeaderboardVO()
{
_players = new Vector.<LeaderboardEntryVO>;
_alliances = new Vector.<LeaderboardEntryVO>;
}

public function update( leaderboardData:LeaderboardData ):void
{
_lastUpdate = getTimer();
_type = leaderboardData.leaderboardType;
var players:Vector.<LeaderboardPlayerData> = leaderboardData.players;
var alliances:Vector.<LeaderboardAllianceData> = leaderboardData.alliances;
var len:uint = players.length;
var i:uint;
var
```

```
currentData:LeaderboardEntryVO;
_players.length = len;
for (; i < len; ++i)
{
currentData = new LeaderboardEntryVO();
currentData.setUpFromPlayerData(players[i]);
_players[i] = currentData;
}

len = alliances.length;
_alliances.length = len;
for (i = 0; i < len; ++i)
{
currentData = new LeaderboardEntryVO();
currentData.setUpFromAllianceData(alliances[i]);
_alliances[i] = currentData;
}
}

public function needsUpdate():Boolean
{
var needsUpdate:Boolean;
if (getTimer() - _lastUpdate > _refreshTime)
needsUpdate = true;

return needsUpdate;
}

public function get players():Vector.<LeaderboardEntryVO>
{
return _players;
}

public function get alliances():Vector.<LeaderboardEntryVO>
{
return _alliances;
}
}
}

------------------------------------
File 324: igw\com\model\mail\MailModel.as
package com.model.mail
{
import com.model.Model;
import com.service.server.incoming.data.MailInboxData;

import org.osflash.signals.Signal;

import flash.utils.Dictionary;

public
```

```
class MailModel extends Model
{
private var _unreadCount:uint;
private var _mailCount:uint;
private var _mail:Vector.<MailVO>;
private var _mailInvites:Dictionary;

public var countUpdated:Signal;
public var mailHeadersUpdated:Signal;
public var mailDetailUpdated:Signal;

[PostConstruct]
public function init():void
{
_mail = new Vector.<MailVO>;
_mailInvites = new Dictionary();
countUpdated = new Signal(uint, uint, Boolean);
mailHeadersUpdated = new Signal(Vector.<MailVO>);
mailDetailUpdated = new Signal(MailVO);
}

public function updateCount( unread:uint, count:uint, serverUpdate:Boolean ):void
{
unreadCount = unread;
mailCount = count;
countUpdated.dispatch(unread, count, serverUpdate);
}

public function addMailHeaders( v:Vector.<MailInboxData> ):void
{
var oldMail:Vector.<MailVO> = _mail.concat();
_mail = new Vector.<MailVO>;
_mailInvites = new Dictionary();

var len:uint = v.length;
var currentMail:MailVO;
var currentMailData:MailInboxData;
for (var i:uint = 0; i < len; ++i)
{
currentMailData = v[i];
currentMail = getMailByKey(currentMailData.key, oldMail);

if (currentMail)
currentMail.updateMailData(currentMailData.sender, currentMailData.subject,
currentMailData.isRead, currentMailData.timeSent);
else if (currentMailData.key != '') //Tell Steve about this bug!
currentMail = new MailVO(currentMailData.key, currentMailData.sender,
currentMailData.subject, currentMailData.isRead, currentMailData.timeSent);

if (currentMail)
{
```

```
_mail.push(currentMail);

// Look for the alliance invitations in the mail subject
// Match strings between Join and alliance
var joinAllianceRegExp:RegExp = /Join.*?alliance/g;
var mailSubjectJoinAllanceParts:Array = currentMail.subject.match(joinAllianceRegExp);

var currentSubjectJoinAlliancePart:String;
for (var j:uint = 0; j < mailSubjectJoinAllanceParts.length; ++j)
{
currentSubjectJoinAlliancePart = mailSubjectJoinAllanceParts[j];
if(currentSubjectJoinAlliancePart.length > 0)
{
// Prepare extracted alliance key to match database key
var allianceKey:String = currentSubjectJoinAlliancePart.replace("Join ", "");
allianceKey = allianceKey.replace(" alliance", "");
allianceKey = "alliance." + allianceKey;
allianceKey = allianceKey.toLowerCase();
allianceKey = allianceKey.replace(" ", "_");
if (!(allianceKey in _mailInvites))
{
_mailInvites[allianceKey] = allianceKey;
}
}
}
}
}
oldMail.length = 0;
mailHeadersUpdated.dispatch(_mail);
}

public function addMailDetail( key:String, sender:String, senderAlliance:String, body:String,
senderRace:String, html:Boolean ):void
{
var mail:MailVO = getMailByKey(key, _mail);
if (mail)
{
mail.addDetail(sender, senderAlliance, body, senderRace, html);
mailDetailUpdated.dispatch(mail);
}
}

public function getMailByKey( key:String, mailHolder:Vector.<MailVO> ):MailVO
{
var mail:MailVO;
if (mailHolder)
{
var len:uint = mailHolder.length;
var currentMail:MailVO;
for
```

```
(var i:uint = 0; i < len; ++i)
{
currentMail = mailHolder[i];
if (currentMail.key == key)
{
mail = currentMail;
break;
}
}

}

return mail;
}

private function deleteMailByKey( key:String ):void
{
if (_mail)
{
var len:uint = _mail.length;
var currentMail:MailVO;
for (var i:uint = 0; i < len; ++i)
{
currentMail = _mail[i];
if (currentMail.key == key)
{
_mail.splice(i, 1);
if (!currentMail.isRead)
{
if (_unreadCount != 0)
_unreadCount -= 1;

updateCount(_unreadCount, _mailCount, false);
}
currentMail = null;
break;
}
}

}
}

public function deleteMail( key:String ):void
{
deleteMailByKey(key);
}

public function mailRead( key:String ):void
{
var mail:MailVO = getMailByKey(key, _mail);
if
```

```
(mail && !mail.isRead)
{
mail.isRead = true;
if (_unreadCount != 0)
_unreadCount -= 1;

updateCount(_unreadCount, _mailCount, false);
}
}

public function set unreadCount( v:uint ):void { _unreadCount = v; }

public function get unreadCount():uint { return _unreadCount; }

public function set mailCount( v:uint ):void { _mailCount = v; }

public function get mailCount():uint { return _mailCount; }

public function get mail():Vector.<MailVO> { return _mail; }

public function getMailInvites():Dictionary { return _mailInvites; }
}
}

-------------------------------------
File 325: igw\com\model\mail\MailVO.as
package com.model.mail
{
public class MailVO
{
private var _key:String;
private var _sender:String;
private var _senderKey:String;
private var _senderAlliance:String;
private var _subject:String;
private var _body:String;
private var _sendersRace:String;
private var _isRead:Boolean;
private var _showHtml:Boolean;
private var _timeSent:Number;

public function MailVO( key:String, sender:String, subject:String, isRead:Boolean,
timeSent:Number )
{
_key = key;
_sender = sender;
_subject = subject;
_isRead = isRead;
_timeSent = timeSent;

}
```

```
public function updateMailData( sender:String, subject:String, isRead:Boolean,
timeSent:Number ):void
{
_sender = sender;
_subject = subject;
_isRead = isRead;
_timeSent = timeSent;
}

public function addDetail( senderKey:String, senderAlliance:String, body:String,
senderRace:String, html:Boolean ):void
{
_senderKey = senderKey;
_senderAlliance = senderAlliance;
_body = body;
_sendersRace = senderRace;
_showHtml = html;
}

public function get key():String { return _key; }
public function get senderKey():String { return _senderKey; }
public function get allianceKey():String { return _senderAlliance; }
public function get sender():String { return _sender; }
public function get subject():String { return _subject; }
public function get body():String { return _body; }
public function get sendersRace():String { return _sendersRace; }
public function get isRead():Boolean { return _isRead; }
public function set isRead( v:Boolean ):void { _isRead = v; }
public function get showHtml():Boolean { return _showHtml; }
public function get timeSent():Number { return _timeSent; }
}
}

-----------------------------------
File 326: igw\com\model\mission\MissionInfoVO.as
package com.model.mission
{
public class MissionInfoVO
{
public var alloyReward:uint;
public var creditReward:uint;
public var energyReward:uint;
public var syntheticReward:uint;
public var palladiumCurrencyReward:uint;

public var blueprintReward:Boolean;

public var currentProgress:int = 0;
public
```

```
var progressRequired:int = 0;

private var _dialogStrings:Array = [];
private var _soundStrings:Array = [];
private var _smallImages:Array = [];
private var _mediumImages:Array = [];
private var _largeImages:Array = [];
private var _npcTitle:Array = [];
private var _objectives:Array = [];
private var _titleColor:uint = 0xffffff;

public function addDialog( v:String ):void { _dialogStrings.push(v); }
public function addSound( v:String ):void { _soundStrings.push(v); }
public function addImages( small:String, medium:String, large:String ):void
{
_smallImages.push(small);
_mediumImages.push(medium);
_largeImages.push(large);
}
public function addObjective( v:String ):void { _objectives.push(v); }
public function addTitle( v:String, color:uint = 0xffffff ):void { _npcTitle.push(v); _titleColor = color;
}

public function get hasDialog():Boolean { return _dialogStrings.length > 0; }
public function get hasSound():Boolean { return _soundStrings.length > 0; }
public function get hasImage():Boolean { return _smallImages.length > 0; }
public function get hasObjectives():Boolean { return _objectives.length > 0; }
public function get hasTitle():Boolean { return _npcTitle.length > 0; }

public function get dialog():String
{
if (_dialogStrings.length > 0)
return _dialogStrings.shift();
return '';
}
public function get sound():String
{
if (_soundStrings.length > 0)
return _soundStrings.shift();
return '';
}

public function get smallImage():String
{
if (_smallImages.length > 0)
{
_mediumImages.unshift();
_largeImages.shift();
return _smallImages.shift();
}
return
```

```
";
}

public function get mediumImage():String
{
if (_mediumImages.length > 0)
{
_smallImages.shift();
_largeImages.shift();
return _mediumImages.shift();
}
return '';
}

public function get largeImage():String
{
if (_largeImages.length > 0)
{
_smallImages.shift();
_mediumImages.shift();
return _largeImages.shift();
}
return '';
}

public function get npcTitle():String
{
if (_npcTitle.length > 0)
return _npcTitle.shift();
return '';
}

public function get objective():String
{
if (_objectives.length > 0)
return _objectives.shift();
return '';
}

public function get titleColor():uint { return _titleColor; }

public function destroy():void
{
currentProgress = 0;
_dialogStrings.length = 0;
_soundStrings.length = 0;
_smallImages.length = 0;
_mediumImages.length = 0;
_largeImages.length = 0;
_npcTitle.length = 0;
_objectives.length
```

```
= 0;
progressRequired = 0;
}
}
}

------------------------------------
File 327: igw\com\model\mission\MissionModel.as
package com.model.mission
{
import com.enum.FactionEnum;
import com.enum.MissionEnum;
import com.model.Model;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.MissionData;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class MissionModel extends Model
{
private var _mission:MissionVO;
private var _missionLookup:Dictionary = new Dictionary(true);
private var _missions:Vector.<MissionVO> = new Vector.<MissionVO>;
private var _missionUpdated:Signal = new Signal();
private var _prototypeModel:PrototypeModel;

public function importMissionData( mission:MissionData ):void
{
if (mission.prototype == null)
return;
if (!_missionLookup.hasOwnProperty(mission.id))
{
var missionVO:MissionVO = ObjectPool.get(MissionVO);
missionVO.init(mission.id);
_missionLookup[mission.id] = missionVO;
_missions.push(missionVO);
if (mission.category == MissionEnum.STORY)
{
_mission = missionVO;
_missionUpdated.dispatch();
}
}
_missionLookup[mission.id].importData(mission);
}

public
```

```
function missionAccepted():void
{
_mission.accepted = true;
}

public function missionComplete():void
{
_mission.progress = _mission.progressRequired;
}

public function missionRewardAccepted():void
{
_mission.rewardAccepted = true;
_missionUpdated.dispatch();
}

public function getStoryMission( chapterID:int, missionID:int ):MissionVO
{
var faction:String = "IGA";
if (CurrentUser.faction != FactionEnum.IGA)
faction = (CurrentUser.faction == FactionEnum.SOVEREIGNTY) ? "SOV" : "TYR";
var id:String = faction + "_Ch" + chapterID + "_M" + missionID;
var proto:IPrototype = _prototypeModel.getMissionPrototye(id);
if (proto)
{
var missionData:MissionData = ObjectPool.get(MissionData);
missionData.prototype = proto;
var missionVO:MissionVO = ObjectPool.get(MissionVO);
missionVO.importData(missionData);
ObjectPool.give(missionData);
return missionVO;
}
return null;
}

public function getMissionByCategory( category:String ):MissionVO
{
for (var i:int = 0; i < _missions.length; i++)
{
if (_missions[i].category == category)
return _missions[i];
}
return null;
}

public function getMissionByID( id:String ):MissionVO
{
if (_missionLookup.hasOwnProperty(id))
return _missionLookup[id];
return null;
}
```

```
public function get currentMission():MissionVO { return _mission; }

public function addListenerToUpdateMission( listener:Function ):void {
_missionUpdated.add(listener); }
public function removeListenerToUpdateMission( listener:Function ):void {
_missionUpdated.remove(listener); }

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
}
}



---
File 328: igw\com\model\mission\MissionVO.as
package com.model.mission
{
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.MissionData;

public class MissionVO implements IPrototype
{
public var accepted:Boolean;
public var progress:int;
public var rewardAccepted:Boolean;
public var sector:String;

private var _chapter:int;
private var _id:String;
private var _mission:int;
private var _prototype:IPrototype;

public function init( id:String ):void
{
_id = id;
}

public function importData( missionData:MissionData ):void
{
accepted = missionData.accepted;
progress = missionData.progress;
rewardAccepted = missionData.rewardAccepted;
sector = missionData.sector;
_prototype = missionData.prototype;
var temp:Array = name.split("_");
_chapter = int(temp[1].substr(2));
_mission = int(temp[2].substr(1));
if
```

```
(_mission < 0)
_mission = 1;
}

public function getUnsafeValue( key:String ):* { return _prototype.getUnsafeValue(key); }
public function getValue( key:String ):* { return _prototype.getValue(key); }

public function get category():String { return _prototype.getValue("category"); }
public function get complete():Boolean { return progress >= progressRequired; }
public function get id():String { return _id; }

public function get briefingDialogue():String { return _prototype.getValue("briefingDialogue"); }
public function get chapter():int { return _chapter; }
public function get failDialogue():String { return _prototype.getValue("failDialogue"); }
public function get greetingDialogue():String { return _prototype.getValue("greetingDialogue"); }
public function get mission():int { return _mission; }
public function get objectives():String { return _prototype.getValue('objectives'); }
public function get progressEvent():String { return _prototype.getValue("progressEvent"); }
public function get progressRequired():int { return _prototype.getValue("progressRequired"); }
public function get situationDialogue():String { return _prototype.getValue("situationDialogue"); }
public function get victoryDialogue():String { return _prototype.getValue("victoryDialogue"); }

public function get asset():String { return _prototype.asset; }
public function get uiAsset():String { return _prototype.uiAsset; }

public function get name():String { return _prototype.name; }
public function get itemClass():String { return _prototype.itemClass; }
public function get buildTimeSeconds():uint { return _prototype.buildTimeSeconds; }

public function get alloyCost():int { return 0; }
public function get creditsCost():int { return 0; }
public function get energyCost():int { return 0; }
public function get syntheticCost():int { return 0; }

public function get alloyReward():int { return _prototype.getValue("alloyReward"); }
public function get creditsReward():int { return _prototype.getValue("creditsReward"); }
public function get energyReward():int { return _prototype.getValue("energyReward"); }
public function get syntheticReward():int { return _prototype.getValue("syntheticReward"); }
public function get palladiumCurrencyReward():int { return
_prototype.getValue("hardCurrencyReward"); }

public function get blueprintReward():Boolean { return _prototype.getValue("blueprintReward"); }

public function get isFTE():Boolean { return name.indexOf("FTE_") == 0; }

public function destroy():void
{
_prototype = null;
}
}
}
```

File 329: igw\com\model\motd\MotDDailyRewardModel.as

```
package com.model.motd
{
import com.model.Model;
import com.service.server.incoming.starbase.StarbaseBaselineResponse;
import com.service.server.incoming.starbase.StarbaseDailyRewardResponse;

import flash.utils.getTimer;

import org.osflash.signals.Signal;

public class MotDDailyRewardModel extends Model
{
private var _escalation:int;
private var _canClaimDelta:Number;
private var _dailyResetsDelta:Number;

private var _header:int;
private var _protocolID:int;

private var _clientTime:int;
private var _temp:Number;
private var _timeMS:Number;
private var _timeRemainingMS:Number;
private var _resetTimeRemainingMS:Number;

private var _rewards:StarbaseDailyRewardResponse;
public var rewardResponse:Signal;

[PostConstruct]
public function init():void
{
rewardResponse = new Signal(StarbaseDailyRewardResponse);
}

public function addData(escalation:int, canClaimDelta:Number, dailyResetsDelta:Number,
header:int, protocol:int):void
{
_escalation = escalation;
_timeRemainingMS = _canClaimDelta = canClaimDelta;
_resetTimeRemainingMS = _dailyResetsDelta = dailyResetsDelta;

_header = header;
_protocolID = protocol;

_clientTime = getTimer();
}

public
```

```
function addRewardData(reward:StarbaseDailyRewardResponse):void
{
_rewards = reward;
if(_rewards)
rewardResponse.dispatch(_rewards);
}

public function get timeRemainingMS():Number
{
_temp = _timeRemainingMS - (getTimer() - _clientTime);
if (_temp < 0)
_temp = 0;
return _temp;
}

public function get resetTimeRemainingMS():Number
{
_temp = _resetTimeRemainingMS - (getTimer() - _clientTime);
if (_temp < 0)
_temp = 0;
return _temp;
}

public function get escalation():int { return _escalation; }
public function set escalation(value:int):void { _escalation = value; }

public function get canClaimDelta():Number { return _canClaimDelta; }
public function set canClaimDelta(value:Number):void { _canClaimDelta = value; }

public function get dailyResetsDelta():Number { return _dailyResetsDelta; }
public function set dailyResetsDelta(value:Number):void { _dailyResetsDelta = value; }

public function get header():int { return _header; }
public function set header(value:int):void { _header = value; }

public function get protocolID():int { return _protocolID; }
public function set protocolID(value:int):void { _protocolID = value; }
}
}
-------------------------------------
File 330: igw\com\model\motd\MotDModel.as
package com.model.motd
{
import com.model.Model;
import com.service.server.incoming.data.MotdData;

import org.osflash.signals.Signal;

public class MotDModel extends Model
{
private
```

```
var _messageCount:uint;
private var _motd:Vector.<MotDVO>;

public var newMessage:Signal;

[PostConstruct]
public function init():void
{
_motd = new Vector.<MotDVO>;
newMessage = new Signal(Vector.<MotDVO>);

}

public function addMessages( v:Vector.<MotdData> ):void
{
var messageVO:MotDVO;
for (var i:int; i < v.length; i++)
{
messageVO = new MotDVO(v[i].key, v[i].imageURL, v[i].isRead, v[i].title, v[i].subtitle, v[i].text,
v[i].startTime);
_motd.push(messageVO);
}

// _motd.sort(orderItems);

if (_motd && _motd.length > 0)
newMessage.dispatch(_motd); //[0]);
}

private function orderItems( itemOne:MotDVO, itemTwo:MotDVO ):Number
{

if (!itemOne)
return -1;
if (!itemTwo)
return 1;

var dateSentOne:Number = itemOne.dateSent;
var dateSent:Number = itemTwo.dateSent;

if (dateSentOne > dateSent)
return -1;
else if (dateSentOne < dateSent)
return 1;

return 0;
}

public function get hasMessage():Boolean { return _motd && _motd.length > 0; }

public
```

```
function get motd():Vector.<MotDVO> { return _motd; }
}
}

------------------------------------
File 331: igw\com\model\motd\MotDVO.as
package com.model.motd
{
public class MotDVO
{
private var _key:String;
private var _imageURL:String;
private var _isRead:Boolean
private var _title:String;
private var _subtitle:String;
private var _message:String;
private var _dateSent:Number;

public function MotDVO(key:String, imageURL:String, isRead:Boolean, title:String,
subtitle:String, message:String, dateSent:Number)
{
_key = key;
_imageURL = imageURL;
_isRead = isRead;
_title = title;
_subtitle = subtitle
_message = message
_dateSent = dateSent;
}

public function get key():String { return _key; }
public function get imageURL():String { return _imageURL; }
public function get isRead():Boolean { return _isRead; }
public function get title():String { return _title; }
public function get subtitle():String{ return _subtitle; }
public function get message():String { return _message; }
public function get dateSent():Number { return _dateSent; }

public function set isRead(value:Boolean):void { _isRead = value; }



}
}
------------------------------------
File 332: igw\com\model\player\BookmarkVO.as
package com.model.player
{
import com.model.prototype.IPrototype;

public
```

```
class BookmarkVO
{
private var _name:String;
private var _sector:String;
private var _sectorPrototype:IPrototype;
private var _sectorNamePrototype:IPrototype;
private var _sectorEnumPrototype:IPrototype;
private var _x:int;
private var _y:int;
private var _index:int;

public function BookmarkVO( name:String, sector:String, sectorPrototype:IPrototype,
sectorNamePrototype:IPrototype, sectorEnumPrototype:IPrototype, x:int, y:int, index:uint )
{
_name = name;
_sector = sector;
_sectorPrototype = sectorPrototype;
_sectorNamePrototype = sectorNamePrototype;
_sectorEnumPrototype = sectorEnumPrototype;
_x = x;
_y = y;
_index = index;
}

public function get name():String { return _name; }
public function set name( v:String ):void { _name = v; }
public function get sector():String { return _sector; }
public function get sectorName():String { return (_sectorNamePrototype != null) ?
_sectorNamePrototype.getValue('nameString') : '' }
public function get sectorEnum():String { return (_sectorEnumPrototype != null) ?
_sectorEnumPrototype.getValue('nameString') : '' }
public function get sectorPrototype():IPrototype { return _sectorPrototype; }
public function get sectorNamePrototype():IPrototype { return _sectorNamePrototype; }
public function get sectorEnumPrototype():IPrototype { return _sectorEnumPrototype; }
public function get x():int { return _x; }
public function get y():int { return _y; }
public function get displayX():String { return String(int(_x * 0.01)); }
public function get displayY():String { return String(int(_y * 0.01)); }
public function get index():int { return _index; }
}
}

------------------------------------
File 333: igw\com\model\player\CurrentUser.as
package com.model.player
{
import com.enum.PlayerUpdateEnum;
import com.model.prototype.IPrototype;

import org.osflash.signals.Signal;

public
```

```
class CurrentUser
{
public static var onPlayerUpdate:Signal = new Signal(int, String, String);
public static var onPlayerOffersUpdated:Signal = new Signal(Vector.<OfferVO>);
public static var onBookmarksUpdated:Signal = new Signal(Vector.<BookmarkVO>);

private static var _bookmarks:Vector.<BookmarkVO> = new Vector.<BookmarkVO>;
private static var _hasPromptedForBuildAction:Boolean = false;
private static var _offers:Vector.<OfferVO> = new Vector.<OfferVO>;
private static var _user:PlayerVO = new PlayerVO(true);
private static var _battleFaction:String = '';

public static function initUser( user:PlayerVO ):void
{
_user = user;
}

public static function get hasPromptedForBuildAction():Boolean { return
_hasPromptedForBuildAction; }
public static function set hasPromptedForBuildAction( value:Boolean ):void {
_hasPromptedForBuildAction = value; }

public static function get id():String { return _user.id; }
public static function set id( v:String ):void { _user.id = v; }

public static function get language():String { return _user.language; }
public static function set language( v:String ):void { _user.language = v; }

public static function get country():String { return _user.country; }
public static function set country( v:String ):void { _user.country = v; }

public static function get name():String { return _user.name; }
public static function set name( v:String ):void
{
var prev:String = _user.name;
_user.name = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_NAME, prev, v);
}

public static function get naid():String { return _user.naid; }
public static function set naid( v:String ):void { _user.naid = v; }

public static function set authID( id:String ):void { _user.authID = id; }
public static function get authID():String { return _user.authID; }

public static function set oAuthID( id:String ):void { _user.oAuthID = id; }
public static function get oAuthID():String { return _user.oAuthID; }

public static function get level():int { return _user.level; }
public static function set level( v:int ):void
{
```

```
var prev:int = _user.level;
_user.level = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_LEVEL, String(prev), String(v));
}

public static function get group():int { return _user.group; }
public static function set group( v:int ):void
{
var prev:int = _user.group;
_user.group = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_GROUP, String(prev), String(v));
}

public static function set battleFaction( v:String ):void
{
_battleFaction = v;
}
public static function get battleFaction():String { return _battleFaction; }

public static function get faction():String { return _user.faction; }
public static function set faction( v:String ):void
{
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_FACTION, String(_user.faction), String(v));
_user.faction = v;
}

public static function get alliance():String { return _user.alliance; }
public static function set alliance( v:String ):void
{
var prev:String = _user.alliance;
_user.alliance = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_ALLIANCE, prev, v);
}


public static function get playerWalletKey():String { return _user.playerWalletKey; }
public static function set playerWalletKey( v:String ):void { _user.playerWalletKey = v; }

public static function get allianceName():String { return _user.allianceName; }
public static function set allianceName( v:String ):void { _user.allianceName = v; }

public static function get allianceRank():int { return _user.allianceRank; }
public static function set allianceRank( v:int ):void { _user.allianceRank = v; }

public static function get allowAllianceInvites():Boolean { return _user.allowAllianceInvites; }
public static function set allowAllianceInvites( v:Boolean ):void { _user.allowAllianceInvites = v; }

public static function get isAllianceOpen():Boolean { return _user.isAllianceOpen; }
public static function set isAllianceOpen( v:Boolean ):void { _user.isAllianceOpen = v; }

public
```

```
static function get centerSpaceBase():String { return _user.centerSpaceBase; }
public static function set centerSpaceBase( v:String ):void { _user.centerSpaceBase = v; }

public static function get homeBase():String { return _user.homeBase; }
public static function set homeBase( v:String ):void { _user.homeBase = v; }

public static function get wallet():PlayerWallet { return _user.wallet; }

public static function get vo():PlayerVO { return _user; }

public static function get xp():int { return _user.xp; }
public static function set xp( v:int ):void
{
var prev:int = _user.xp;
_user.xp = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_XP, String(prev), String(v));
}

public static function set commendationPointsPVE( v:uint ):void {
_user.commendationPointsPVE = v; }
public static function get commendationPointsPVE():uint { return
_user.commendationPointsPVE; }

public static function set commendationPointsPVP( v:uint ):void {
_user.commendationPointsPVP = v; }
public static function get commendationPointsPVP():uint { return
_user.commendationPointsPVP; }

public static function get avatarName():String { return _user.avatarName; }
public static function set avatarName( v:String ):void { _user.avatarName = v; }

public static function get baseRating():int { return _user.baseRating; }
public static function set baseRating( v:int ):void
{
var prev:int = _user.baseRating;
_user.baseRating = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_BASERATING, String(prev), String(v));
}

public static function get wins():uint { return _user.wins; }
public static function set wins( v:uint ):void { _user.wins = v; }

public static function get losses():uint { return _user.losses; }
public static function set losses( v:uint ):void { _user.losses = v; }

public static function get draws():uint { return _user.draws; }
public static function set draws( v:uint ):void { _user.draws = v; }


public static function get offers():Vector.<OfferVO> { return _offers; }
public
```

```
static function set offers( v:Vector.<OfferVO> ):void
{
if (_offers.length > 0)
_offers.length = 0;

_offers = v;

onPlayerOffersUpdated.dispatch(_offers);
}

public static function get user():PlayerVO { return _user; }

public static function removeOffer( v:String ):void
{
if (_offers.length > 0)
{
var len:uint = _offers.length;
var currentOffer:OfferVO;
for (var i:uint = 0; i < len; ++i)
{
currentOffer = _offers[i];
if (currentOffer.offerPrototype == v)
{
_offers.splice(i, 1);
currentOffer = null;
currentOffer
break;
}
}
onPlayerOffersUpdated.dispatch(_offers);
}
}

public static function addBookmark( name:String, sector:String, sectorPrototype:IPrototype,
sectorNamePrototype:IPrototype, sectorEnumPrototype:IPrototype, x:int, y:int, index:int ):void
{
var bookmark:BookmarkVO = new BookmarkVO(name, sector, sectorPrototype,
sectorNamePrototype, sectorEnumPrototype, x, y, index);
_bookmarks.push(bookmark);
}

public static function addBookmarks( v:Vector.<BookmarkVO> ):void
{
_bookmarks.length = 0;
_bookmarks = v;
}

public static function removeBookmark( v:uint ):void
{
if (_bookmarks.length > 0)
{
```

```
var len:uint = _bookmarks.length;
var currentBookmark:BookmarkVO;
for (var i:uint = 0; i < len; ++i)
{
currentBookmark = _bookmarks[i];
if (currentBookmark.index == v)
{
_bookmarks.splice(i, 1);
currentBookmark = null;
break;
}
}
}
}

public static function get bookmarks():Vector.<BookmarkVO> { return _bookmarks; }

public static function get bookmarkCount():int
{
if (_bookmarks)
return _bookmarks.length;
else
return 0;
}

public static function get nextBookmarkIndex():uint
{
var index:uint;
var len:uint = _bookmarks.length;
if (len > 0)
{
var currentBookmark:BookmarkVO = _bookmarks[len - 1];
index = currentBookmark.index + 1;
} else
index = 1;
return index;
}

public static function get purchasedShipSlots():int { return _user.purchasedShipSlots; }
public static function set purchasedShipSlots( v:int ):void
{
var prev:int = _user.purchasedShipSlots;
_user.purchasedShipSlots = v;
onPlayerUpdate.dispatch(PlayerUpdateEnum.TYPE_PURCHASED_SHIP_SLOTS,
String(prev), String(v));
}
}
}

-----------------------------------
```

File 334: igw\com\model\player\OfferVO.as

```
package com.model.player
{
public class OfferVO
{
private var _offerPrototype:String;
private var _offerDuration:Number;
private var _endTimeStamp:Number;
private var _clientTime:Number;
private var _timeRemaining:Number;

public function OfferVO( offerPrototype:String, offerDuration:Number, endTimeStamp:Number )
{
_offerPrototype = offerPrototype;
_offerDuration = offerDuration;
_endTimeStamp = endTimeStamp;
}

public function get offerDuration():Number { return _offerDuration; }
public function get endTimestamp():Number { return _endTimeStamp; }
public function get offerPrototype():String { return _offerPrototype; }

public function get timeRemainingMS():Number
{
var currentTime:Date = new Date();
_timeRemaining = _endTimeStamp - currentTime.time;
if (_timeRemaining < 0)
_timeRemaining = 0;
return _timeRemaining;
}
}
}
```

File 335: igw\com\model\player\PlayerModel.as

```
package com.model.player
{
import com.model.Model;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class PlayerModel extends Model
{
private var _players:Dictionary;

public var onPlayerAdded:Signal = new Signal(PlayerVO);

public
```

```
function PlayerModel()
{
_players = new Dictionary();
}

public function addPlayer( player:PlayerVO ):void
{
_players[player.id] = player;

onPlayerAdded.dispatch(player);
}

public function addPlayers( v:Vector.<PlayerVO> ):void
{
var len:uint = v.length;
var currentPlayer:PlayerVO;
for (var i:uint = 0; i < len; ++i)
{
currentPlayer = v[i];
_players[currentPlayer.id] = currentPlayer;
onPlayerAdded.dispatch(currentPlayer);
}
}

public function getPlayer( id:String ):PlayerVO
{
if(id == "")
return null;

if (id in _players)
return _players[id];

return null;
}

public function getPlayers():Dictionary
{
return _players;
}

public function removePlayer( id:String ):void
{
if (id in _players)
{
ObjectPool.give(_players[id]);
delete _players[id];
}
}

public function removeAllPlayers():void
{
```

```
for (var player:String in _players)
{
ObjectPool.give(_players[player]);
delete _players[player];
}
}

public function destroy():void
{
removeAllPlayers();
}
}
}

-----------------------------------
File 336: igw\com\model\player\PlayerVO.as
package com.model.player
{
import com.service.server.BinaryInputStream;
import com.util.CommonFunctionUtil;

public class PlayerVO
{
private var _alliance:String = '';
private var _allianceName:String = '';
private var _allianceRank:int = -1;
private var _allowAllianceInvites:Boolean = true;
private var _isAllianceOpen:Boolean = true;
private var _avatarName:String = '';
private var _authID:String; //kabamID used for authorization with the servers
private var _oAuthID:String; //OAuthID needed for payment!
private var _centerspace:int = -1;
private var _centerSpaceBase:String;
private var _faction:String = '';
private var _group:int = -1;
private var _homeBase:String;
private var _id:String; //id on the game servers
private var _isNPC:Boolean;
private var _level:int = 1;
private var _language:String = '';
private var _country:String = '';
private var _naid:String = ''; //chat id
private var _name:String;
private var _baseRating:int = 0;
private var _wallet:PlayerWallet;
private var _xp:int;
private var _commendationPointsPVE:uint = 0;
private var _commendationPointsPVP:uint = 0;
private var _wins:uint = 0;
private
```

```
var _losses:uint = 0;
private var _draws:uint = 0;
private var _lastOnline:uint = 0;
private var _baseSector:String = '';
private var _baseXPos:Number = 0;
private var _baseYPos:Number = 0;
private var _purchasedShipSlots:int = 0;
private var _playerWalletKey:String = '';

public function PlayerVO( isCurrentUser:Boolean = false )
{
if (isCurrentUser)
_wallet = new PlayerWallet();
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();
id = input.readUTF();
name = input.readUTF();
faction = input.readUTF();
_avatarName = input.readUTF();
_xp = input.readInt64();
_isNPC = input.readBoolean();
_level = 1;
_alliance = input.readUTF(); // allianceKey
_allianceName = input.readUTF(); // allianceName
_purchasedShipSlots = input.readInt64();
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get alliance():String { return _alliance; }
public function set alliance( value:String ):void { _alliance = value; }

public function get allianceName():String { return _allianceName; }
public function set allianceName( value:String ):void { _allianceName = value; }

public function get allianceRank():int { return _allianceRank; }
public function set allianceRank( value:int ):void { _allianceRank = value; }

public function get allowAllianceInvites():Boolean { return _allowAllianceInvites; }
public function set allowAllianceInvites( value:Boolean ):void { _allowAllianceInvites = value; }

public function set authID( id:String ):void { _authID = id; }
public function get authID():String { return _authID; }

public
```

```
function set oAuthID( id:String ):void { _oAuthID = id; }
public function get oAuthID():String { return _oAuthID; }

public function get centerSpaceBase():String { return _centerSpaceBase; }
public function set centerSpaceBase( value:String ):void { _centerSpaceBase = value; }

public function get faction():String { return _faction; }
public function set faction( value:String ):void { _faction = value; }

public function get group():int { return _group; }
public function set group( value:int ):void { _group = value; }

public function get homeBase():String { return _homeBase; }
public function set homeBase( value:String ):void { _homeBase = value; }

public function get id():String { return _id; }
public function set id( v:String ):void { _id = v; }

public function get isNPC():Boolean { return _isNPC; }

public function get level():int { if (_level == 1) _level = CommonFunctionUtil.findPlayerLevel(_xp);
return _level; }
public function set level( value:int ):void { _level = value; }

public function set language( value:String ):void { _language = value; }
public function get language():String { return _language; }

public function set country( value:String ):void { _country = value; }
public function get country():String { return _country; }

public function set naid( naid:String ):void { _naid = naid; }
public function get naid():String { return _naid; }

public function set name( name:String ):void { _name = name; }
public function get name():String { return _name; }

public function get wallet():PlayerWallet { return _wallet; }

public function get xp():int { return _xp; }
public function set xp( value:int ):void { _xp = value; }

public function set commendationPointsPVE( v:uint ):void { _commendationPointsPVE = v; }
public function get commendationPointsPVE():uint { return _commendationPointsPVE; }

public function set commendationPointsPVP( v:uint ):void { _commendationPointsPVP = v; }
public function get commendationPointsPVP():uint { return _commendationPointsPVP; }

public function get avatarName():String { return _avatarName; }
public function set avatarName( value:String ):void { _avatarName = value; }

public
```

```
function get baseRating():int { return _baseRating; }
public function set baseRating( v:int ):void { _baseRating = v; }

public function get isAllianceOpen():Boolean { return _isAllianceOpen; }
public function set isAllianceOpen( v:Boolean ):void { _isAllianceOpen = v; }

public function get wins():uint { return _wins; }
public function set wins( v:uint ):void { _wins = v; }

public function get losses():uint { return _losses; }
public function set losses( v:uint ):void { _losses = v; }

public function get draws():uint { return _draws; }
public function set draws( v:uint ):void { _draws = v; }

public function get lastOnline():uint { return _lastOnline; }
public function set lastOnline( v:uint ):void { _lastOnline = v; }

public function get baseSector():String { return _baseSector; }
public function set baseSector( v:String ):void { _baseSector = v; }

public function get baseXPos():Number { return _baseXPos; }
public function set baseXPos( v:Number ):void { _baseXPos = v; }

public function get baseYPos():Number { return _baseYPos; }
public function set baseYPos( v:Number ):void { _baseYPos = v; }

public function get purchasedShipSlots():int { return _purchasedShipSlots; }
public function set purchasedShipSlots( v:int ):void { _purchasedShipSlots = v; }

public function get playerWalletKey():String { return _playerWalletKey; }
public function set playerWalletKey( v:String ):void { _playerWalletKey = v; }



public function destroy():void
{

}
}
}

-----------------------------------
File 337: igw\com\model\player\PlayerWallet.as
package com.model.player
{
import com.enum.CurrencyEnum;

import org.osflash.signals.Signal;

public
```

```
class PlayerWallet
{
public var onPremiumChange:Signal = new Signal(Boolean);

private var _premium:uint = 0;
private var _prevPremium:uint = 0;

public function deposit( amount:uint, type:String ):void
{
switch (type)
{
case CurrencyEnum.PREMIUM:
_prevPremium = amount;
_premium += amount;
onPremiumChange.dispatch(true);
break;
}
}

public function withdraw( amount:uint, type:String ):Boolean
{
switch (type)
{
case CurrencyEnum.PREMIUM:
if (_premium < amount)
return false;
_prevPremium = amount;
_premium -= amount;
onPremiumChange.dispatch(false);
break;
}
return true;
}

public function getPrevAddedAmount( type:String ):uint
{
switch (type)
{
case CurrencyEnum.PREMIUM:
return _prevPremium;
}
return 0;
}

public function getAmount( type:String ):uint
{
switch (type)
{
case CurrencyEnum.PREMIUM:
return _premium;
}
```

```
return 0;
}

public function get premium():uint { return _premium; }

public function set overridePremium( v:uint ):void { if (_premium != v) _premium = v;
onPremiumChange.dispatch(false); }
}
}

-----------------------------------
File 338: igw\com\model\prototype\IPrototype.as
package com.model.prototype
{
public interface IPrototype
{
function getValue( key:String ):*;
function getUnsafeValue( key:String ):*;

function get asset():String;
function get uiAsset():String;

function get name():String;
function get itemClass():String;
function get buildTimeSeconds():uint;

function get alloyCost():int;
function get creditsCost():int;
function get energyCost():int;
function get syntheticCost():int;
}
}

-----------------------------------
File 339: igw\com\model\prototype\PrototypeModel.as
package com.model.prototype
{
import com.model.Model;
import com.model.asset.AssetModel;

import flash.utils.Dictionary;

public class PrototypeModel extends Model
{
public static var instance:PrototypeModel;

public var attachPoints:Dictionary;
public var attachGroups:Dictionary;
public var moduleProtoTypes:Dictionary;
public
```

```
var moduleGroupPrototypes:Dictionary;

public static var SHIP_REPAIR_TIME_SCALAR:Number;
public static var SHIP_REPAIR_RESOURCE_COST_SCALAR:Number;
public static var SHIP_BUILD_TIME_SCALAR:Number;
public static var SHIP_BUILD_RESOURCE_COST_SCALAR:Number;
public static var BUILDING_BUILD_TIME_SCALAR:Number;
public static var BUILDING_RESOURCE_COST_SCALAR:Number;

private var _assetModel:AssetModel;

private var _blueprintPrototypes:Dictionary;
private var _buffPrototypes:Dictionary;
private var _buildingPrototypes:Dictionary;
private var _cachedQuery:Dictionary;
private var _missionObjectivesPrototypes:Dictionary;
private var _missionPrototypes:Dictionary;
private var _researchRequirements:Dictionary;
private var _slotPrototypes:Dictionary;
private var _constantPrototypes:Dictionary;
private var _sectorPrototypes:Dictionary;
private var _sectorNamePrototypes:Dictionary;
private var _splitTestCohortPrototypes:Dictionary;
private var _offerPrototypes:Dictionary;
private var _npcPrototypes:Dictionary;
private var _statPrototypes:Dictionary;
private var _commendationRankPrototypes:Dictionary;
private var _splitTestPrototypes:Dictionary;

private var _loadingScreenPrototypes:Dictionary;
private var _vLoadingScreenPrototypes:Vector.<IPrototype>;

private var _loadingScreenGroupPrototypes:Dictionary;
private var _vLoadingScreenGroupPrototypes:Vector.<IPrototype>;

private var _vBuffPrototypes:Vector.<IPrototype>;
private var _vBuildingPrototypes:Vector.<IPrototype>;
private var _vBuildableBuildingPrototype:Vector.<IPrototype>;
private var _vFTESteps:Vector.<IPrototype>;
private var _vShipPrototypes:Vector.<IPrototype>;
private var _vFirstNamePrototypes:Vector.<IPrototype>;
private var _vLastNamePrototypes:Vector.<IPrototype>;
private var _vBEDialoguePrototypes:Vector.<IPrototype>;
private var _vLoginBonusPrototypes:Vector.<IPrototype>;
private var _vEventPrototypes:Vector.<IPrototype>;

private var _researchPrototypes:Dictionary;
private var _vResearchPrototypes:Vector.<IPrototype>;

private var _statmodPrototypes:Dictionary;
private
```

```
var _vStatmodPrototypes:Vector.<IPrototype>;

private var _weaponPrototypes:Dictionary;
private var _vWeaponPrototypes:Vector.<IPrototype>;

private var _transgateCustomDestinationPrototypes:Dictionary;
private var _vTransgateCustomDestinationPrototypes:Vector.<IPrototype>;

private var _transgateCustomDestinationGroupPrototypes:Dictionary;
private var _vTransgateCustomDestinationGroupPrototypes:Vector.<IPrototype>;

private var _factionPrototypes:Dictionary;
private var _vFactionPrototypes:Vector.<IPrototype>

private var _contractPrototypes:Dictionary;
private var _vContractPrototypes:Vector.<IPrototype>;

private var _agentPrototypes:Dictionary;
private var _vAgentPrototypes:Vector.<IPrototype>;

private var _dialogPrototypes:Dictionary;
private var _vDialogPrototypes:Vector.<IPrototype>;

private var _csRacePrototypes:Dictionary;
private var _vCsRacePrototypes:Vector.<IPrototype>;

private var _storeItemPrototypes:Dictionary;
private var _vStoreItemPrototypes:Vector.<IPrototype>;

private var _offerItemPrototypes:Dictionary;
private var _vOfferItemPrototypes:Vector.<IPrototype>;

private var _faqEntryPrototypes:Dictionary;
private var _vFAQEntryPrototypes:Vector.<IPrototype>;

private var _achievementPrototypes:Dictionary;
private var _vAchievementPrototypes:Vector.<IPrototype>;

private var _filterAchievementPrototypes:Dictionary;
private var _vFilterAchievementPrototypes:Vector.<IPrototype>;

private var _debuffPrototypes:Dictionary;
private var _vDebuffPrototypes:Vector.<IPrototype>;

private var _activeSplits:Vector.<IPrototype>;

private const TOAST_CONGRATS:String = "CodeString.Toast.Congratulations"; //
Congratulations!
private const TOAST_YOU_WILL_RECEIVE:String = "CodeString.Toast.YouWillReceive"; //
YOU WILL RECEIVE:

[PostConstruct]
```

```
public function init():void
{


attachPoints = new Dictionary();
attachGroups = new Dictionary();
_blueprintPrototypes = new Dictionary();
_buffPrototypes = new Dictionary();
_vBuffPrototypes = new Vector.<IPrototype>;
_buildingPrototypes = new Dictionary();
_vBuildingPrototypes = new Vector.<IPrototype>;
_vBuildableBuildingPrototype = new Vector.<IPrototype>;
_debuffPrototypes = new Dictionary();
_vDebuffPrototypes = new Vector.<IPrototype>;
_cachedQuery = new Dictionary();
_vFTESteps = new Vector.<IPrototype>;
_missionPrototypes = new Dictionary();
_missionObjectivesPrototypes = new Dictionary();
modulePrototypes = new Dictionary();
moduleGroupPrototypes = new Dictionary();
_researchPrototypes = new Dictionary();
_vResearchPrototypes = new Vector.<IPrototype>;
_researchRequirements = new Dictionary();
_vShipPrototypes = new Vector.<IPrototype>;
_slotPrototypes = new Dictionary();
_constantPrototypes = new Dictionary();
_statmodPrototypes = new Dictionary();
_vStatmodPrototypes = new Vector.<IPrototype>;
_weaponPrototypes = new Dictionary();
_vWeaponPrototypes = new Vector.<IPrototype>;
_transgateCustomDestinationPrototypes = new Dictionary();
_vTransgateCustomDestinationPrototypes = new Vector.<IPrototype>;
_transgateCustomDestinationGroupPrototypes = new Dictionary();
_vTransgateCustomDestinationGroupPrototypes = new Vector.<IPrototype>;
_factionPrototypes = new Dictionary();
_vFactionPrototypes = new Vector.<IPrototype>;
_contractPrototypes = new Dictionary();
_vContractPrototypes = new Vector.<IPrototype>;
_agentPrototypes = new Dictionary();
_vAgentPrototypes = new Vector.<IPrototype>;
_dialogPrototypes = new Dictionary();
_vDialogPrototypes = new Vector.<IPrototype>;
_csRacePrototypes = new Dictionary();
_vCsRacePrototypes = new Vector.<IPrototype>;
_vFirstNamePrototypes = new Vector.<IPrototype>;
_vLastNamePrototypes = new Vector.<IPrototype>;
_sectorPrototypes = new Dictionary();
_sectorNamePrototypes = new Dictionary();
_splitTestCohortPrototypes = new Dictionary();
_storeItemPrototypes
```

```
= new Dictionary();
_vStoreItemPrototypes = new Vector.<IPrototype>;
_npcPrototypes = new Dictionary();
_offerPrototypes = new Dictionary();
_offerItemPrototypes = new Dictionary();
_vOfferItemPrototypes = new Vector.<IPrototype>;
_statPrototypes = new Dictionary();
_vBEDialoguePrototypes = new Vector.<IPrototype>;
_faqEntryPrototypes = new Dictionary();
_vFAQEntryPrototypes = new Vector.<IPrototype>;
_vFAQEntryPrototypes = new Vector.<IPrototype>;
_vLoginBonusPrototypes = new Vector.<IPrototype>;
_commendationRankPrototypes = new Dictionary();
_vAchievementPrototypes = new Vector.<IPrototype>;
_achievementPrototypes = new Dictionary;
_vFilterAchievementPrototypes = new Vector.<IPrototype>;
_filterAchievementPrototypes = new Dictionary;
_vEventPrototypes = new Vector.<IPrototype>;
_splitTestPrototypes = new Dictionary();
_loadingScreenPrototypes = new Dictionary();
_vLoadingScreenPrototypes = new Vector.<IPrototype>;

_loadingScreenGroupPrototypes = new Dictionary();
_vLoadingScreenGroupPrototypes = new Vector.<IPrototype>;

_activeSplits = new Vector.<IPrototype>;

instance = this;

var steps:Array = [
{
name:"START",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"showTipModal",
timeDelay:20,
missionName:"",
anchor:false,
mood:0,
stepId:"300000"
},
{
name:"Combat Intro",
platform:"Browser",
dialogString:"FTE.CombatIntro",
voID:"sounds/vo/fte/FTE_001.mp3",
uiID:"",
```

cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"moveToFleet",
timeDelay:0,
missionName:"",
anchor:false,
mood:2,
stepId:"300001"
},
{
name:"Combat Initiated",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"forceMissionComplete,disableNext",
timeDelay:0,

missionName:"FTE_IGA_Starting_Mission,FTE_SOV_Starting_Mission,FTE_TYR_Starting_Mission",
anchor:false,
mood:0,
stepId:"300002"
},
{
name:"Combat Initiated",
platform:"Browser",
dialogString:"FTE.CombatInitiated",
voID:"sounds/vo/fte/FTE_002.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"moveToFleet,checkFleetInBattle",
timeDelay:5,
missionName:"",
anchor:false,
mood:3,
stepId:"301002"
},
{
name:"Combat Initiated",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.hud.sector.SectorView",
cutoutCoordinates:"0,100,640,40",
arrowCoordinates:"320,144,-90",
trigger:"",
timeDelay:0,

missionName:"",
anchor:false,
mood:3,
stepId:"301003"
},
{
name:"Combat Movement 2",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"hideDialogue,hideOverlay,progressOnStateChange",
timeDelay:0,
missionName:"",
anchor:false,
mood:3,
stepId:"301004"
},
{
name:"Combat Movement",
platform:"Browser",
dialogString:"FTE.CombatSelection",
voID:"sounds/vo/fte/FTE_003.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"pauseBattle,centerOnFleets,disableNext,pressWASD,toastImage|WASD_Keys.png",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"301005"
},
{
name:"Combat Movement 2",
platform:"Browser",
dialogString:"FTE.FiringRange",
voID:"sounds/vo/fte/FTE_004.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"moveToFleet,disableNext,selectFleet,killToast",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"301006"
},

{
name:"Combat Movement 3",
platform:"Browser",
dialogString:"",
voID:"sounds/vo/fte/VO_FTE_Combat_003_Click_To_Move.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"centerOnFleets,waitForMove,disableNext,pointToCenterOfScreen",
timeDelay:0,
missionName:"",
anchor:false,
mood:1,
stepId:"301007"
},
{
name:"Combat Active",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"centerOnFleets,unpauseBattle,hideDialogue,hideOverlay,notifyOnBattleEnd",
timeDelay:0,
missionName:"",
anchor:true,
mood:1,
stepId:"301008"
},
{
name:"Combat End",
platform:"Browser",
dialogString:"FTE.CombatEnd",
voID:"sounds/vo/fte/FTE_005.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"301009"
},
{
name:"Looting/Rewards",
platform:"Browser",
dialogString:"FTE.LootingRewards",
voID:"sounds/vo/fte/FTE_006.mp3",

uiID:"com.ui.modal.battle.BattleEndView",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"highlightUI|lootHolder",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"301010"
},
{
name:"Combat Targeting 3",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.battle.BattleEndView",
cutoutCoordinates:"762,463,138,38",
arrowCoordinates:"819,460,90",
trigger:"hideDialogue",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"301011"
},
{
name:"Combat Targeting 3",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"hideOverlay,progressOnStateChange",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"301012"
},
{
name:"Substep",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:0,

missionName:"FTE_IGA_Dock,FTE_SOV_Dock,FTE_TYR_Dock",
anchor:false,
mood:0,
stepId:"301013"
},
{
name:"Recall",
platform:"Browser",
dialogString:"FTE.Recall",
voID:"sounds/vo/fte/FTE_007.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"moveToFleet,selectFleet,disableNext",
timeDelay:5,
missionName:"",
anchor:false,
mood:0,
stepId:"302014"
},
{
name:"Substep",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.core.component.contextmenu.ContextMenu",
cutoutCoordinates:"15,33,122,17",
arrowCoordinates:"0,40,0",
trigger:"selectContextMenu,followFleet",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"302015"
},
{
name:"Substep",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"hideDialogue,checkForMission",
timeDelay:0,

missionName:"FTE_TYR_Upgrade_Shipyard_Begin,FTE_IGA_Upgrade_Shipyard_Begin,FTE_SOV_Upg
anchor:false,
mood:0,
stepId:"302016"

},
{
name:"Enter Base",
platform:"Browser",
dialogString:"FTE.EnterBase",
voID:"sounds/vo/fte/FTE_008.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"selectBase,disableNext",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"303017"
},
{
name:"Substep",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.core.component.contextmenu.ContextMenu",
cutoutCoordinates:"15,33,122,17",
arrowCoordinates:"0,40,0",
trigger:"selectContextMenu",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"303018"
},
{
name:"Substep",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"progressOnStateChange,hideDialogue,hideOverlay",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"303019"
},
{
name:"Greeting",
platform:"Browser",
dialogString:"FTE.Greeting",

voID:"sounds/vo/fte/FTE_009.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"303020"
},
{
name:"Base Intro",
platform:"Browser",
dialogString:"FTE.RewardIntro",
voID:"sounds/vo/fte/FTE_010.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"highlightUI|HardCurrency",
timeDelay:0,
missionName:"",
anchor:true,
mood:1,
stepId:"303021"
},
{
name:"Shipyard Upgrade",
platform:"Browser",
dialogString:"FTE.ShipyardIntro",
voID:"sounds/vo/fte/FTE_011.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"disableNext,selectBuilding|ConstructionBay",
timeDelay:5,
missionName:"",
anchor:false,
mood:0,
stepId:"303022"
},
{
name:"Subtask",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.core.component.contextmenu.ContextMenu",
cutoutCoordinates:"14,70,122,20",
arrowCoordinates:"0,80,0",
trigger:"selectContextMenu",

timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"303023"
},
{
name:"Shipyard Upgrade 1 Click",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.construction.ConstructionInfoView",
cutoutCoordinates:"401,464,119,50",
arrowCoordinates:"460,470,90",
trigger:"hideDialogue",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"303024"
},
{
name:"Interlude",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:0,

missionName:"FTE_TYR_Upgrade_Shipyard,FTE_IGA_Upgrade_Shipyard,FTE_SOV_Upgrade_Shipyard
anchor:false,
mood:0,
stepId:"303025"
},
{
name:"Shipyard Upgrade 2",
platform:"Browser",
dialogString:"FTE.ShipyardInstant",
voID:"sounds/vo/fte/FTE_012.mp3",
uiID:"com.ui.hud.shared.PlayerView",
cutoutCoordinates:"238,81,126,53",
arrowCoordinates:"290,134,-90",
trigger:"",
timeDelay:5,
missionName:"",
anchor:false,
mood:0,

stepId:"304026"
},
{
name:"Shipyard Upgrade 4 Click",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.store.StoreView",
cutoutCoordinates:"310,430,395,85",
arrowCoordinates:"500,440,90",
trigger:"hideDialogue",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"304027"
},
{
name:"Subtask",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"closeView|com.ui.modal.store.StoreView,forceNextStep",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"304028"
},
{
name:"Interlude",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:5,

missionName:"FTE_TYR_Build_Ship_Begin,FTE_IGA_Build_Ship_Begin,FTE_SOV_Build_Ship_Begin",
anchor:false,
mood:0,
stepId:"304029"
},
{
name:"Trade

Routes Intro",
platform:"Browser",
dialogString:"FTE.Palladium",
voID:"sounds/vo/fte/FTE_013.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"highlightUI|HardCurrency",
timeDelay:5,
missionName:"",
anchor:false,
mood:0,
stepId:"305030"
},
{
name:"Ship Intro",
platform:"Browser",
dialogString:"FTE.ShipBuildIntro",
voID:"sounds/vo/fte/FTE_014.mp3",
uiID:"com.ui.hud.shared.engineering.EngineeringView",
cutoutCoordinates:"322,0,155,41",
arrowCoordinates:"395,32,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305031"
},
{
name:"Ship Intro 2",
platform:"Browser",
dialogString:"FTE.HullIntro",
voID:"sounds/vo/fte/FTE_015.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305032"
},
{
name:"Weapon Slot",
platform:"Browser",
dialogString:"FTE.WeaponSlot",
voID:"sounds/vo/fte/FTE_016.mp3",
uiID:"com.ui.modal.shipyard.ShipyardView",
cutoutCoordinates:"",

arrowCoordinates:"",
trigger:"selectShipSlot|Weapon",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305033"
},
{
name:"Weapon Slot Select",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.construction.ConstructionView",
cutoutCoordinates:"280,83,590,113",
arrowCoordinates:"280,123,0",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305034"
},
{
name:"Defense Slot",
platform:"Browser",
dialogString:"FTE.DefenseSlot",
voID:"sounds/vo/fte/FTE_017.mp3",
uiID:"com.ui.modal.shipyard.ShipyardView",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"selectShipSlot|Defense",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305035"
},
{
name:"Defense Slot 02",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.construction.ConstructionView",
cutoutCoordinates:"280,83,590,113",
arrowCoordinates:"280,123,0",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,

```
mood:0,
stepId:"305036"
},
{
name:"Power Capacity",
platform:"Browser",
dialogString:"FTE.PowerCapacity",
voID:"sounds/vo/fte/FTE_018.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305037"
},
{
name:"Ship Build",
platform:"Browser",
dialogString:"FTE.ShipBuild",
voID:"sounds/vo/fte/FTE_019.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"305038"
},
{
name:"Ship Build Click",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.shipyard.ShipyardView",
cutoutCoordinates:"551,549,120,52",
arrowCoordinates:"611,550,90",
trigger:"hideDialogue",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"305039"
},
{
name:"Subtask",
```

platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"closeView|com.ui.modal.shipyard.ShipyardView,checkForMission,disableNext",
timeDelay:0,
missionName:"FTE_TYR_Build_Ship,FTE_IGA_Build_Ship,FTE_SOV_Build_Ship",
anchor:false,
mood:2,
stepId:"305040"
},
{
name:"Subtask",
platform:"Browser",
dialogString:"FTE.Palladium2",
voID:"sounds/vo/fte/FTE_020.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"",
timeDelay:5,
missionName:"",
anchor:false,
mood:2,
stepId:"306041"
},
{
name:"Shipyard Accelerate 01",
platform:"Browser",
dialogString:"FTE.DashboardIntro",
voID:"sounds/vo/fte/FTE_021.mp3",
uiID:"com.ui.hud.shared.engineering.EngineeringView",
cutoutCoordinates:"343,46,114,13",
arrowCoordinates:"395,57,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"306042"
},
{
name:"Shipyard Accelerate 02",
platform:"Browser",
dialogString:"FTE.DashboardIntro2",
voID:"sounds/vo/fte/FTE_022.mp3",
uiID:"com.ui.hud.shared.engineering.EngineeringView",
cutoutCoordinates:"336,95,60,60",

arrowCoordinates:"365,143,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:0,
stepId:"306042"
},
{
name:"Shipyard Accelerate 03",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.store.StoreView",
cutoutCoordinates:"308,108,400,90",
arrowCoordinates:"508,102,90",
trigger:"enableHUD",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"306044"
},
{
name:"Subtask",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"closeView|com.ui.modal.store.StoreView,checkForMission,disableNext",
timeDelay:0,

missionName:"FTE_TYR_Repair_Fleet_Begin,FTE_IGA_Repair_Fleet_Begin,FTE_SOV_Repair_Fleet_Be
anchor:false,
mood:0,
stepId:"306045"
},
{
name:"Ship Complete",
platform:"Browser",
dialogString:"FTE.Kabam",
voID:"sounds/vo/fte/FTE_023.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"closeView|com.ui.modal.shipyard.ShipyardView",
timeDelay:5,
missionName:"",

anchor:true,
mood:3,
stepId:"307046"
},
{
name:"Fleet Tab",
platform:"Browser",
dialogString:"FTE.Messing",
voID:"sounds/vo/fte/FTE_024.mp3",
uiID:"com.ui.hud.shared.engineering.EngineeringView",
cutoutCoordinates:"483,0,155,41",
arrowCoordinates:"558,32,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:1,
stepId:"307047"
},
{
name:"Repair Fleet",
platform:"Browser",
dialogString:"FTE.FleetIntro",
voID:"sounds/vo/fte/FTE_025.mp3",
uiID:"com.ui.modal.dock.DockView",
cutoutCoordinates:"700,260,118,50",
arrowCoordinates:"760,310,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"307048"
},
{
name:"Subtask",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:0,
missionName:"FTE_TYR_Repair_Fleet,FTE_IGA_Repair_Fleet,FTE_SOV_Repair_Fleet",
anchor:false,
mood:0,
stepId:"307049"
},
{

name:"Click speed up",
platform:"Browser",
dialogString:"FTE.FleetIntro",
voID:"",
uiID:"com.ui.modal.dock.DockView",
cutoutCoordinates:"683,260,118,50",
arrowCoordinates:"750,310,-90",
trigger:"",
timeDelay:5,
missionName:"",
anchor:false,
mood:0,
stepId:"308050"
},
{
name:"Repair Fleet 2",
platform:"Browser",
dialogString:"FTE.InstantReminder",
voID:"sounds/vo/fte/FTE_026.mp3",
uiID:"com.ui.modal.store.StoreView",
cutoutCoordinates:"305,108,404,90",
arrowCoordinates:"507,102,90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:false,
mood:2,
stepId:"308051"
},
{
name:"Repair Fleet 2",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"31,135,225,35",
arrowCoordinates:"584,150,90",
trigger:"checkForMission,disableNext",
timeDelay:0,
missionName:"FTE_TYR_Update_Fleet,FTE_IGA_Update_Fleet,FTE_SOV_Update_Fleet",
anchor:false,
mood:0,
stepId:"308052"
},
{
name:"Repair Fleet 2",
platform:"Browser",
dialogString:"FTE.ProgressPanel",
voID:"sounds/vo/fte/FTE_027.mp3",
uiID:"com.ui.modal.store.StoreView",

cutoutCoordinates:"31,135,225,35",
arrowCoordinates:"143,130,90",
trigger:"ignoreStoreOffset",
timeDelay:5,
missionName:"",
anchor:true,
mood:0,
stepId:"309053"
},
{
name:"Fleet slots",
platform:"Browser",
dialogString:"FTE.LaunchFleet",
voID:"sounds/vo/fte/FTE_028.mp3",
uiID:"com.ui.modal.dock.DockView",
cutoutCoordinates: "157,400,115,105",//"265,220,115,105",
arrowCoordinates: "215,398,90",//"315,318,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"309054"
},
{
name:"Fleet slots 02",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"com.ui.modal.dock.ShipSelectionView",
cutoutCoordinates:"22,135,590,105",
arrowCoordinates:"315,244,-90",
trigger:"",
timeDelay:0,
missionName:"",
anchor:true,
mood:2,
stepId:"309055"
},
{
name:"Switch to Sector View",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:0,
missionName:"FTE_TYR_Launch_Fleet,FTE_IGA_Launch_Fleet,FTE_SOV_Launch_Fleet",

anchor:false,
mood:0,
stepId:"309056"
},
{
name:"Launch Fleet",
platform:"Browser",
dialogString:"",
voID:"sounds/vo/fte/VO_FTE_Main_010_Tiger.mp3",
uiID:"com.ui.modal.dock.DockView",
cutoutCoordinates:"731,585,233,38",
arrowCoordinates:"868,591,90",
trigger:"hideDialogue",
timeDelay:5,
missionName:"",
anchor:true,
mood:2,
stepId:"310057"
},
{
name:"Switch to Sector View",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"hideDialogue,hideOverlay,progressOnStateChange",
timeDelay:0,
missionName:"",
anchor:true,
mood:0,
stepId:"310058"
},
{
name:"Repair Fleet 2",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:0,
missionName:"FTE_TYR_Reward,FTE_IGA_Reward,FTE_SOV_Reward",
anchor:false,
mood:0,
stepId:"411060"
},
{

name:"Transgate Intro 2",
platform:"Browser",
dialogString:"FTE.FleetButton",
voID:"sounds/vo/fte/FTE_029.mp3",
uiID:"com.ui.hud.shared.command.CommandView",
cutoutCoordinates:"163,-66,186,60",
arrowCoordinates:"160,-40,0",
trigger:"forceMissionComplete,disableNext",
timeDelay:10,
missionName:"",
anchor:true,
mood:0,
stepId:"411060"
},
{
name:"AI Intro",
platform:"Browser",
dialogString:"FTE.Reward",
voID:"sounds/vo/fte/FTE_030.mp3",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"moveToFleet",
timeDelay:0,
missionName:"",
anchor:true,
mood:1,
stepId:"411061"
},
{
name:"Switch to Sector View",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"checkForMission,disableNext",
timeDelay:0,

missionName:"TYR_Ch1_M1,IGA_Ch1_M1,SOV_Ch1_M1,TYR_Ch0_M1,IGA_Ch0_M1,SOV_Ch0_M1,T
anchor:false,
mood:0,
stepId:"411062"
},
{
name:"Combat Initiated",
platform:"Browser",
dialogString:"FTE.GoodLuck",
voID:"sounds/vo/fte/FTE_031.mp3",

```
uiID:"com.ui.hud.shared.bridge.BridgeView",
cutoutCoordinates:"12,130,60,60",
arrowCoordinates:"100,154,180",
trigger:"disableNext",
timeDelay:5,
missionName:"",
anchor:true,
mood:2,
stepId:"411063"
},
{
name:"END",
platform:"Browser",
dialogString:"",
voID:"",
uiID:"",
cutoutCoordinates:"",
arrowCoordinates:"",
trigger:"forceNextStep",
timeDelay:20,
missionName:"",
anchor:false,
mood:0,
stepId:"411064"
}
];


for (var i:int = 0; i < steps.length; i++)
{
_vFTESteps.push(new PrototypeVO(steps[i]));
}
}

public function addPrototypeData( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
}
}
}

/*
Returns a loading screen prototype by its key.
*/
public function getLoadingScreenPrototype(key:String):IPrototype{
return
```

```
_loadingScreenPrototypes[key];
}

/*
Returns a vector list containing all loading screen prototypes.
*/
public function getLoadingScreenPrototypes():Vector.<IPrototype>{
return _vLoadingScreenPrototypes;
}

/*
Returns a vector list containing all loading screen prototypes by group.
*/
public function getLoadingScreenPrototypesByGroup(group:String):Vector.<IPrototype>{
var loadingScreenGroupPrototypes:Vector.<IPrototype> = new Vector.<IPrototype>;

for each (var loadingScreenPrototype:IPrototype in _loadingScreenPrototypes)
{
if (loadingScreenPrototype.getValue('loadingScreenGroup') == group)
loadingScreenGroupPrototypes.push(loadingScreenPrototype);
}
return loadingScreenGroupPrototypes;
}


/*
Returns a loading screen group prototype by its key *Note* this is different than
LoadingScreenPrototypes?.
*/
public function getLoadingScreenGroupPrototype(key:String):IPrototype{
return _loadingScreenPrototypes[key];
}

/*
Returns a vector list containing all loading screen group prototypes *Note* this is different than
LoadingScreenPrototypes or LoadingScreenPrototype Groups?.
*/
public function getLoadingScreenGroupPrototypes():Vector.<IPrototype>{
return _vLoadingScreenGroupPrototypes;
}


public function getBlueprintPrototype( key:String ):IPrototype { return _blueprintPrototypes[key];
}
public function getBuffPrototype( key:String ):IPrototype { return _buffPrototypes[key]; }
public function getBuffPrototypes():Vector.<IPrototype> { return _vBuffPrototypes; }

public function getDebuffPrototype( key:String ):IPrototype { return _debuffPrototypes[key]; }
public function getDebuffPrototypes():Vector.<IPrototype> { return _vDebuffPrototypes; }

public
```

```
function getBuildingPrototype( key:String ):IPrototype
{
return _buildingPrototypes[key];
}
public function getBuildingPrototypeByClassAndLevel( buildingClass:String, level:int ):IPrototype
{
for (var i:int = 0; i < _vBuildingPrototypes.length; i++)
{
if (_vBuildingPrototypes[i].getValue('itemClass') == buildingClass &&
_vBuildingPrototypes[i].getValue('level') == level)
return _vBuildingPrototypes[i];
}
return null;
}

public function getBuildableBuildingPrototypes():Vector.<IPrototype> { return
_vBuildableBuildingPrototype; }
public function getBuildingPrototypes():Vector.<IPrototype> { return _vBuildingPrototypes; }
public function getAttachPoint( key:String ):IPrototype { return attachPoints[key]; }
public function getAttachGroup( key:String ):Array { return attachGroups[key]; }

public function getAttachPointByType( attachGroup:String, attachPointType:String
):PrototypeVO
{
var attachPointSet:Array = attachGroups[attachGroup];
for each (var attachPoint:PrototypeVO in attachPointSet)
{
if (attachPoint.getValue('attachPointType') == attachPointType)
return attachPoint;
}
return null;
}

public function getFTEStepsByPlatform( platform:String ):Vector.<IPrototype>
{
if (!_cachedQuery['FTE' + platform])
{
var steps:Vector.<IPrototype> = new Vector.<IPrototype>;
for (var i:int = 0; i < _vFTESteps.length; i++)
{
if (_vFTESteps[i].getValue("platform") == platform)
steps.push(_vFTESteps[i]);
}
_cachedQuery['FTE' + platform] = steps;
}
return _cachedQuery['FTE' + platform];
}

public function getMissionPrototye( id:String ):IPrototype { return _missionPrototypes[id]; }
public function getMissionObjective( id:String ):IPrototype { return
_missionObjectivesPrototypes[id];
```

```
}

public function getResearchPrototypeByName( key:String ):IPrototype { return
_researchPrototypes[key]; }
public function getResearchPrototypes():Vector.<IPrototype> { return _vResearchPrototypes; }
public function getResearchPrototypesByBuilding( buildingClass:String ):Vector.<IPrototype> { if
(_researchPrototypes.hasOwnProperty(buildingClass)) return
_researchPrototypes[buildingClass]; return null; }
public function getResearchPrototypesDict():Dictionary { return _researchPrototypes; }

public function getStatModPrototypeByName( key:String ):IPrototype { return
_statmodPrototypes[key]; }
public function getStatModPrototypes():Vector.<IPrototype> { return _vStatmodPrototypes; }
public function getStatModPrototypesByGroup( group:String ):Vector.<IPrototype>
{
var mods:Vector.<IPrototype> = new Vector.<IPrototype>;
for each (var statMod:IPrototype in _statmodPrototypes)
{
if (statMod.getValue('modGroup') == group)
mods.push(statMod);
}

return mods;
}

public function getConstantPrototypeByName( key:String ):IPrototype { return
_constantPrototypes[key]; }
public function getConstantPrototypeValueByName( key:String ):* { return
_constantPrototypes[key].getValue('value'); }
public function getConstantPrototypes():Dictionary { return _constantPrototypes; }

public function getFactionPrototypeByName( key:String ):IPrototype { return
_factionPrototypes[key]; }
public function getFactionPrototypes():Vector.<IPrototype> { return _vFactionPrototypes }

public function getContractPrototypeByName( key:String ):IPrototype { return
_contractPrototypes[key]; }
public function getContractPrototypes():Vector.<IPrototype> { return _vContractPrototypes; }

public function getAgentPrototypeByName( key:String ):IPrototype { return
_agentPrototypes[key]; }
public function getAgentPrototypes():Vector.<IPrototype> { return _vAgentPrototypes; }

public function getDialogPrototypeByName( key:String ):IPrototype { return
_dialogPrototypes[key]; }
public function getDialogPrototypes():Vector.<IPrototype> { return _vDialogPrototypes; }

public function getNPCPrototypeByName( key:String ):IPrototype { return _npcPrototypes[key]; }
public function getOfferPrototypeByName( key:String ):IPrototype { return _offerPrototypes[key];
}

public
```

```
function getStatPrototypeByName( key:String ):IPrototype { return _statPrototypes[key]; }

public function getOfferItemsByItemGroup( key:String ):Vector.<IPrototype>
{
var groupItems:Vector.<IPrototype> = new Vector.<IPrototype>;

for (var i:int = 0; i < _vOfferItemPrototypes.length; i++)
{
if (_vOfferItemPrototypes[i].getValue('itemGroup') == key)
groupItems.push(_vOfferItemPrototypes[i]);
}
return groupItems;
}

public function getSectorPrototypeByName( key:String ):IPrototype { return
_sectorPrototypes[key]; }

public function getSectorNamePrototypeByName( key:String ):IPrototype { return
_sectorNamePrototypes[key]; }

public function getSplitTestCohortPrototypeByName( key:String ):IPrototype { return
_splitTestCohortPrototypes[key]; }

public function getStoreItemPrototypeByName( key:String ):IPrototype { return
_storeItemPrototypes[key]; }
public function getStoreItemPrototypes():Vector.<IPrototype> { return _vStoreItemPrototypes; }

public function getShipPrototype( type:String ):IPrototype
{
for (var i:int = 0; i < _vShipPrototypes.length; i++)
{
if (_vShipPrototypes[i].name == type)
return _vShipPrototypes[i];
}

return null;
}
public function getShipPrototypesByFaction( faction:String ):Vector.<IPrototype>
{
var ships:Vector.<IPrototype> = new Vector.<IPrototype>;
for (var i:int = 0; i < _vShipPrototypes.length; i++)
{
if (_vShipPrototypes[i].getValue('faction') == faction)
ships.push(_vShipPrototypes[i]);
}

return ships;
}
public function getSlotPrototype( key:String ):IPrototype { return _slotPrototypes[key]; }
public function getModulesBySlotType( slotType:String ):Vector.<IPrototype>
{
```

```
var key:String = "shipComponent" + slotType;
if (_cachedQuery[key])
return _cachedQuery[key];
var modules:Vector.<IPrototype> = new Vector.<IPrototype>;
for each (var vo:IPrototype in _weaponPrototypes)
{
if (vo.getValue('slotType') == slotType)
modules.push(vo);
}
_cachedQuery[key] = modules;
return modules;
}

public function getWeaponPrototype( key:String ):IPrototype { return _weaponPrototypes[key]; }

public function getWeaponPrototypes():Vector.<IPrototype> { return _vWeaponPrototypes; }

public function getTransgateCustomDestinationPrototype( key:String ):IPrototype { return
_transgateCustomDestinationPrototypes[key]; }

public function getTransgateCustomDestinationGroupByCustomDestinationGroup( group:String
):Vector.<IPrototype>
{
var key:String = "transgateCustomDestination" + group;
if (_cachedQuery[key])
return _cachedQuery[key];
var customDestinations:Vector.<IPrototype> = new Vector.<IPrototype>;
for each (var vo:IPrototype in _transgateCustomDestinationGroupPrototypes)
{
if (vo.getValue('transgateCustomDestinationGroup') == group)
customDestinations.push(vo);
}
_cachedQuery[key] = customDestinations;
return customDestinations;
}

public function getRacePrototypesByFaction( faction:String, race:String ):Vector.<IPrototype>
{
var races:Vector.<IPrototype> = new Vector.<IPrototype>;
var currentRace:IPrototype;
for (var i:int = 0; i < _vCsRacePrototypes.length; i++)
{
currentRace = _vCsRacePrototypes[i];
if (currentRace.getValue('faction') == faction && currentRace.getValue('race') == race &&
currentRace.getValue('isPc') == true && currentRace.getValue('isActive') == true)
races.push(currentRace);
}
return races;
}

public
```

```
function getRacePrototypeByName( key:String ):IPrototype { return _csRacePrototypes[key]; }

public function getFirstNamePrototypes():Vector.<IPrototype> { return _vFirstNamePrototypes; }
public function getLastNamePrototypes():Vector.<IPrototype> { return _vLastNamePrototypes; }

public function getBEDialogueByFaction( faction:String, result:String = 'Victory'
):Vector.<IPrototype>
{
var options:Vector.<IPrototype> = new Vector.<IPrototype>;
//Result is either Victory or Taunt
for each (var option:IPrototype in _vBEDialoguePrototypes)
{
if (option.getValue('faction') == faction && option.getValue('resultType') == result)
options.push(option);
}
return options;
}

public function getFirstNameOptions( race:String, gender:String ):Vector.<IPrototype>
{
//Total hack until I sort the data out
if (race == "Ares Magna")
race = "AresMagna";
var options:Vector.<IPrototype> = new Vector.<IPrototype>;
for each (var name:IPrototype in _vFirstNamePrototypes)
{
if (name.getValue('race') == race && (name.getValue('gender') == gender ||
name.getValue('gender') == 'Unisex'))
options.push(name);
}
return options;
}

public function getLastNameOptions( race:String, gender:String ):Vector.<IPrototype>
{
//Total hack until I sort the data out
if (race == "Ares Magna")
race = "AresMagna";
var options:Vector.<IPrototype> = new Vector.<IPrototype>;
for each (var name:IPrototype in _vLastNamePrototypes)
{
if (name.getValue('race') == race && (name.getValue('gender') == gender ||
name.getValue('gender') == 'Unisex'))
options.push(name);
}
return options;
}

public function getFAQEntryPrototypesByName( key:String ):IPrototype
{
return
```

```
_faqEntryPrototypes[key];
}

public function getFAQEntryPrototypes():Vector.<IPrototype>
{
return _vFAQEntryPrototypes;
}

public function getLoginBonusPrototypeByDay( day:int ):IPrototype
{
for each (var p:IPrototype in _vLoginBonusPrototypes)
{
if (p.getValue('escalationReq') == day)
return p;
}

return null;
}

public function getCommendationRankPrototypesByName( key:String ):IPrototype
{
return _commendationRankPrototypes[key];
}

public function getAchievementPrototypes():Vector.<IPrototype>
{
return _vAchievementPrototypes;
}

public function getFilterAchievementPrototypes():Vector.<IPrototype>
{
return _vFilterAchievementPrototypes;
}

public function getAchievementPrototypeByName( name:String ):IPrototype
{
return _achievementPrototypes[name];
}

public function getSplitTestPrototypeByName( name:String ):IPrototype
{
return _splitTestPrototypes[name];
}

public function set AttachPointPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
attachPoints[key]
```

```
= vo;
var group:String = vo.getValue("attachGroup");
attachGroups[group] ||= [];
attachGroups[group].push(vo);
}
}

public function set ActiveDefensePrototypes( v:Object ):void
{
WeaponPrototypes = v;
}

public function set BlueprintPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_blueprintPrototypes[key] = vo;
}
}

public function getEventPrototypes():Vector.<IPrototype>
{
return _vEventPrototypes;
}

public function set EventPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vEventPrototypes.push(vo);
}
}

public function set SplitTestPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_splitTestPrototypes[key] = vo;
}

}

public function set BuffPrototypes( v:Object ):void
{
var
```

```
vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_buffPrototypes[key] = vo;
_vBuffPrototypes.push(vo);
}
}

public function set DebuffPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_debuffPrototypes[key] = vo;
_vDebuffPrototypes.push(vo);
}
}

public function set BuildingPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vBuildingPrototypes.push(vo);
_buildingPrototypes[key] = vo;

if (vo.getValue('level') == 1)
_vBuildableBuildingPrototype.push(vo);
}
}

public function set AssetPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_assetModel.addGameAssetData(vo);
}
}

public function set MissionPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_missionPrototypes[key]
```

```
= vo;
}
}

public function set ModulePrototypes( v:Object ):void
{
WeaponPrototypes = v;
// TODO - Delete this maybe? This just seems to break things when run...
}

public function set ModuleGroupPrototypes( v:Object ):void
{

}

public function set ObjectivesPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_missionObjectivesPrototypes[key] = vo;
}
}

public function set ShipPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vShipPrototypes.push(vo);
}
}

public function set SlotPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_slotPrototypes[key] = vo;
}
}

public function set StatModPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo
```

```
= new PrototypeVO(v[key]);
_statmodPrototypes[key] = vo;
}
}

public function set ConstantPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_constantPrototypes[key] = vo;
}
updateScalars();
}

public function set StoreItemPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_storeItemPrototypes[key] = vo;
_vStoreItemPrototypes.push(vo);
}
}

public function set FactionPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_factionPrototypes[key] = vo;
_vFactionPrototypes.push(vo);
}
}

public function set ContractPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_contractPrototypes[key] = vo;
_vContractPrototypes.push(vo);
}
}

public function set AgentPrototypes( v:Object ):void
{
```

```
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_agentPrototypes[key] = vo;
_vAgentPrototypes.push(vo);
}
}

public function set DialogPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_dialogPrototypes[key] = vo;
_vDialogPrototypes.push(vo);
}
}

public function set WeaponPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_weaponPrototypes[key] = vo;
_vWeaponPrototypes.push(vo);
}
}

public function set TransgateCustomDestinationPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_transgateCustomDestinationPrototypes[key] = vo;
_vTransgateCustomDestinationPrototypes.push(vo);
}
}

public function set TransgateCustomDestinationGroupPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_transgateCustomDestinationGroupPrototypes[key] = vo;
_vTransgateCustomDestinationGroupPrototypes.push(vo);
```

```
}
}

public function set RacePrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_csRacePrototypes[key] = vo;
_vCsRacePrototypes.push(vo);
}
}

public function set UIAssetPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_assetModel.addUIAssetData(vo);
}
}

public function set AudioAssetPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_assetModel.addAudioAssetData(vo);
}
}

public function set FilterAssetPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_assetModel.addFilterAssetData(vo);
}
}

public function set FilterAchievementPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo
```

```
= new PrototypeVO(v[key]);
_filterAchievementPrototypes[key] = vo;
_vFilterAchievementPrototypes.push(vo);
}
}

public function set ResearchPrototypes( v:Object ):void
{
var buildingClass:String;
var vectorKey:String;
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
buildingClass = vo.getValue('requiredBuildingClass');
vectorKey = buildingClass + vo.getValue("filterCategory");
_researchPrototypes[key] = vo;
if (!_researchPrototypes.hasOwnProperty(buildingClass))
_researchPrototypes[buildingClass] = new Vector.<IPrototype>;
if (!_researchPrototypes.hasOwnProperty(vectorKey))
_researchPrototypes[vectorKey] = new Vector.<IPrototype>;
_researchPrototypes[buildingClass].push(vo);
_researchPrototypes[vectorKey].push(vo);
_vResearchPrototypes.push(vo);

var requiredTech:String = vo.getValue("requiredResearch");
if (requiredTech.length > 0)
{
_researchRequirements[requiredTech] ||= [];
_researchRequirements[requiredTech].push(key);
}
}
}

public function set CommendationRankPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_commendationRankPrototypes[key] = vo;
}
}

public function getResearchThatRequires( tech:String ):Vector.<IPrototype>
{
var result:Vector.<IPrototype> = new Vector.<IPrototype>();
var keys:Array = _researchRequirements[tech];
if (keys)
{
for
```

```
each (var key:String in keys)
{
result.push(getResearchPrototypeByName(key));
}
}
return result;
}

public function set FirstNamePrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vFirstNamePrototypes.push(vo);
}
}

public function set LastNamePrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vLastNamePrototypes.push(vo);
}
}

public function set FTEPrototypes( v:Object ):void
{
/*var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_fteSteps.push(vo);
}
_fteSteps.sort(sortByID);*/
}

public function set NPCPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_npcPrototypes[key] = vo;
}
}

public function set OfferPrototypes( v:Object ):void
{
```

```
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_offerPrototypes[key] = vo;
}
}

public function set OfferItemPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_offerItemPrototypes[key] = vo;
_vOfferItemPrototypes.push(vo);
}
}

public function set FAQEntryPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_faqEntryPrototypes[key] = vo;
_vFAQEntryPrototypes.push(vo);
}
}

public function set SectorPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_sectorPrototypes[key] = vo;
}
}

public function set AchievementPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_achievementPrototypes[key] = vo;
_vAchievementPrototypes.push(vo);
}
}
```

```
public function set SectorNamePrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_sectorNamePrototypes[key] = vo;
}
}

public function set SplitTestCohortPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_splitTestCohortPrototypes[key] = vo;
}
}

public function set StatPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_statPrototypes[key] = vo;
}
}

public function set BEDialoguePrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vBEDialoguePrototypes.push(vo);
}
}

public function set LoginBonusPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_vLoginBonusPrototypes.push(vo);
}
}
```

```
/*
Sets our loading screen prototype Dictionary Map and Vector List
*/
public function set LoadingScreenPrototypes( v:Object ):void
{
//trace(new Error().getStackTrace());
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_loadingScreenPrototypes[key] = vo;
_vLoadingScreenPrototypes.push(vo);
}
}

/*
Sets our loading screen group prototype Dictionary Map and Vector List *Note* this is different
from
loadingScreenPrototypes?
*/
public function set LoadingScreenGroupPrototypes( v:Object ):void
{
var vo:IPrototype;
for (var key:String in v)
{
vo = new PrototypeVO(v[key]);
_loadingScreenGroupPrototypes[key] = vo;
_vLoadingScreenGroupPrototypes.push(vo);
}
}

private function sortByID( protoA:IPrototype, protoB:IPrototype ):int
{
if (protoA.getValue("id") < protoB.getValue("id"))
return -1;
return 1;
}

public function setSplits( v:Vector.<String> ):void
{
var len:uint;
var i:uint;
var key:String;
var currentSplitTestPrototype:IPrototype;
var prototype:PrototypeVO;
if (_activeSplits)
{
len = _activeSplits.length;
for
```

```
(; i < len; ++i)
{
currentSplitTestPrototype = _activeSplits[i];
key = currentSplitTestPrototype.getValue('targetKey');
prototype = getPrototypeVO(currentSplitTestPrototype.getValue('targetClass'), key);
prototype.removeOverridenValue(key);
}
_activeSplits.length = 0;
}

len = v.length;
var value:*;
for (i = 0; i < len; ++i)
{
//SPLIT
currentSplitTestPrototype = getSplitTestPrototypeByName(v[i]);
key = currentSplitTestPrototype.getValue('targetKey');
value = currentSplitTestPrototype.getValue('valueStr');
if (value == null || value == '')
value = currentSplitTestPrototype.getValue('valueFloat');

prototype = getPrototypeVO(currentSplitTestPrototype.getValue('targetClass'), key);
prototype.overrideValue(currentSplitTestPrototype.getValue('targetColumn'), value);
_activeSplits.push(currentSplitTestPrototype);
}
updateScalars();
}

private function getPrototypeVO( prototypeClass:String, key:String ):PrototypeVO
{
switch (prototypeClass)
{
case 'AchievementPrototype':
if (_achievementPrototypes.hasOwnProperty(key))
return _achievementPrototypes[key];

break;
case 'FilterAchievementPrototype':
if (_filterAchievementPrototypes.hasOwnProperty(key))
return _filterAchievementPrototypes[key];

break;
case 'AgentPrototype':
if (_agentPrototypes.hasOwnProperty(key))
return _agentPrototypes[key];

break;
case 'BlueprintPrototype':
if (_blueprintPrototypes.hasOwnProperty(key))
return _blueprintPrototypes[key];

break;
```

```
case 'BuffPrototype':
if (_buffPrototypes.hasOwnProperty(key))
return _buffPrototypes[key];

break;
case 'BuildingPrototype':
if (_buildingPrototypes.hasOwnProperty(key))
return _buildingPrototypes[key];

break;
case 'DebuffPrototype':
if (_debuffPrototypes.hasOwnProperty(key))
return _debuffPrototypes[key];

break;
case 'ConstantPrototype':
if (_constantPrototypes.hasOwnProperty(key))
return _constantPrototypes[key];

break;
case 'ContractPrototype':
if (_contractPrototypes.hasOwnProperty(key))
return _contractPrototypes[key];

break;
case 'CommendationRankPrototype':
if (_commendationRankPrototypes.hasOwnProperty(key))
return _commendationRankPrototypes[key];

break;
case 'FactionPrototype':
if (_factionPrototypes.hasOwnProperty(key))
return _factionPrototypes[key];

break;
case 'MissionPrototype':
if (_missionPrototypes.hasOwnProperty(key))
return _missionPrototypes[key];

break;
case 'NPCPrototype':
if (_npcPrototypes.hasOwnProperty(key))
return _npcPrototypes[key];

break;
case 'ObjectivesPrototype':
if (_missionObjectivesPrototypes.hasOwnProperty(key))
return _missionObjectivesPrototypes[key];

break;
```

```
case 'OfferItemPrototype':
if (_offerItemPrototypes.hasOwnProperty(key))
return _offerItemPrototypes[key];

break;
case 'OfferPrototype':
if (_offerPrototypes.hasOwnProperty(key))
return _offerPrototypes[key];

break;
case 'RacePrototype':
if (_csRacePrototypes.hasOwnProperty(key))
return _csRacePrototypes[key];

break;
case 'ResearchPrototype':
if (_researchPrototypes.hasOwnProperty(key))
return _researchPrototypes[key];

break;
case 'SectorNamePrototype':
if (_sectorNamePrototypes.hasOwnProperty(key))
return _sectorNamePrototypes[key];

break;
case 'SplitTestCohortPrototype':
if (_splitTestCohortPrototypes.hasOwnProperty(key))
return _splitTestCohortPrototypes[key];

break;
case 'SectorPrototype':
if (_sectorPrototypes.hasOwnProperty(key))
return _sectorPrototypes[key];

break;
case 'ShipPrototypes':
for (var i:uint = 0; i < _vShipPrototypes.length; ++i)
{
if (_vShipPrototypes[i].name == key)
return PrototypeVO(_vShipPrototypes[i]);
}

break;
case 'SlotPrototype':
if (_slotPrototypes.hasOwnProperty(key))
return _slotPrototypes[key];

break;
case 'StatModPrototype':
if
```

```
(_statmodPrototypes.hasOwnProperty(key))
return _statmodPrototypes[key];

break;
case 'StatPrototype':
if (_statPrototypes.hasOwnProperty(key))
return _statPrototypes[key];

break;
case 'StoreItemPrototype':
if (_storeItemPrototypes.hasOwnProperty(key))
return _storeItemPrototypes[key];

break;
case 'WeaponPrototype':
case 'ActiveDefensePrototype':
case 'ModulePrototype':
if (_weaponPrototypes.hasOwnProperty(key))
return _weaponPrototypes[key];

break;
}

return null;
}

private function updateScalars():void
{
SHIP_REPAIR_TIME_SCALAR = getConstantPrototypeValueByName('shipRepairTimeScalar');
SHIP_REPAIR_RESOURCE_COST_SCALAR =
getConstantPrototypeValueByName('shipRepairResourceCostScalar');
SHIP_BUILD_TIME_SCALAR = getConstantPrototypeValueByName('shipBuildTimeScalar');
SHIP_BUILD_RESOURCE_COST_SCALAR =
getConstantPrototypeValueByName('shipBuildResourceCostScalar');
BUILDING_BUILD_TIME_SCALAR =
getConstantPrototypeValueByName('buildingBuildTimeScalar');
BUILDING_RESOURCE_COST_SCALAR =
getConstantPrototypeValueByName('buildingResourceCostScalar');
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
}
}



-----------------------------------
File 340: igw\com\model\prototype\PrototypeVO.as
package com.model.prototype
{
```

```
public class PrototypeVO implements IPrototype
{
private var _data:Object;
private var _overriddenData:Object;

public function PrototypeVO( data:Object )
{
_data = data;
}

public function getValue( key:String ):*
{
if (_overriddenData && _overriddenData.hasOwnProperty(key))
return _overriddenData[key];

if (_data.hasOwnProperty(key))
return _data[key];
else
throw new Error("Parameter " + key + " not found on " + this);
}

public function getUnsafeValue( key:String ):*
{
if (_overriddenData && _overriddenData.hasOwnProperty(key))
return _overriddenData[key];

if (_data.hasOwnProperty(key))
return _data[key];

return null;
}

public function overrideValue( key:String, newValue:* ):void
{
if (_overriddenData == null)
_overriddenData = new Object();

_overriddenData[key] = newValue;
}

public function removeOverridenValue( key:String ):void
{
delete _overriddenData[key];
}

public function get asset():String { return getUnsafeValue('asset'); }
public function get uiAsset():String { return getUnsafeValue('uiAsset'); }

public function get name():String { return getValue('key'); }
public
```

```
function get itemClass():String { return getValue('itemClass'); }
public function get buildTimeSeconds():uint { return getUnsafeValue('buildTimeSeconds'); }

public function get alloyCost():int
{
if (_data.hasOwnProperty('alloyCost'))
return getValue('alloyCost');
return 0;
}
public function get creditsCost():int
{
if (_data.hasOwnProperty('creditsCost'))
return getValue('creditsCost');
return 0;
}
public function get energyCost():int
{
if (_data.hasOwnProperty('energyCost'))
return getValue('energyCost');
return 0;
}
public function get syntheticCost():int
{
if (_data.hasOwnProperty('syntheticCost'))
return getValue('syntheticCost');
return 0;
}
}
}

-------------------------------------
File 341: igw\com\model\scene\SceneModel.as
package com.model.scene
{
import com.model.Model;

import flash.geom.Point;
import flash.geom.Rectangle;

import org.parade.util.DeviceMetrics;

public class SceneModel extends Model
{
private var _bounds:Rectangle;
private var _focus:Point;
private var _ready:Boolean;
private var _viewArea:Rectangle;
private var _zoom:Number;

[PostConstruct]
public
```

```
function init():void
{
_focus = new Point();
_ready = false;
_viewArea = new Rectangle(0, 0, DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS);
_zoom = 1;
}

public function buildScene( width:Number, height:Number ):void
{
_bounds = new Rectangle(0, 0, width, height);
//set the default focus to the center of the play area
setFocus(_bounds.width / 2, _bounds.height / 2);
_ready = true;
}

public function adjustFocus( dx:Number, dy:Number ):void
{
_focus.x += dx;
_focus.y += dy;

//attempt to center
var w2:Number = DeviceMetrics.WIDTH_PIXELS * .5 / _zoom;
var h2:Number = DeviceMetrics.HEIGHT_PIXELS * .5 / _zoom;
_viewArea.x = _focus.x - w2;
_viewArea.y = _focus.y - h2;
if (_viewArea.x < 0)
{
_focus.x = w2;
_viewArea.x = 0;
}
if (_viewArea.y < 0)
{
_focus.y = h2;
_viewArea.y = 0;
}
if (_viewArea.right >= _bounds.width)
{
var ox:Number = _viewArea.right - _bounds.width + 1;
_focus.x -= ox;
_viewArea.x -= ox;
}
if (_viewArea.bottom >= _bounds.height)
{
var oy:Number = _viewArea.bottom - _bounds.height + 1;
_focus.y -= oy;
_viewArea.y -= oy;
}
}

public
```

```
function setFocus( dx:Number, dy:Number ):void
{
_focus.x = dx;
_focus.y = dy;
adjustFocus(0, 0);
}

public function changeResolution():void
{
_viewArea.setTo(_viewArea.x, _viewArea.y, DeviceMetrics.WIDTH_PIXELS / _zoom,
DeviceMetrics.HEIGHT_PIXELS / _zoom);
if (_ready)
adjustFocus(0, 0);
}

public function get bounds():Rectangle { return _bounds; }
public function get focus():Point { return _focus; }
public function get ready():Boolean { return _ready; }
public function get viewArea():Rectangle { return _viewArea; }
public function get zoom():Number { return _zoom; }
public function set zoom( v:Number ):void { _zoom = v; changeResolution(); }

public function cleanup():void
{
_ready = false;
_bounds = null;
zoom = 1;
}
}
}

-------------------------------------
File 342: igw\com\model\sector\SectorModel.as
package com.model.sector
{
import com.model.Model;
import com.model.prototype.IPrototype;
import com.service.language.Localization;
import com.service.server.incoming.data.SectorData;

import flash.geom.Point;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class SectorModel extends Model
{
public var focusLocation:Point = new Point();
public var targetSector:String;
public
```

```
var viewBase:Boolean = false;

private var _focusFleetID:String; //used when going to a new sector to specify which fleet to focus on when the player gets there

private var _destinations:Vector.<SectorVO>;
private var _privateDestinations:Vector.<SectorVO>;
private var sectorIDToSector:Dictionary;
private var _sectorChangeSignal:Signal;
private var _selectedFleetIDChanged:Signal;
private var _sector:SectorVO = new SectorVO();

private var _igaContextMenuDefaultIndex:int = -1;
private var _tyrContextMenuDefaultIndex:int = -1;
private var _sovContextMenuDefaultIndex:int = -1;
private var _csContextMenuDefaultIndex:int = -1;


[PostConstruct]
public function init():void
{
_sectorChangeSignal = new Signal(String);
_selectedFleetIDChanged = new Signal();
sectorIDToSector = new Dictionary();
}

public function updateSector( sector:SectorData ):void
{
_sector.importData(sector);
_sectorChangeSignal.dispatch(sector.id);
}

public function addDestinations( sectors:Vector.<SectorData> ):void
{
sectorIDToSector = new Dictionary();
//clear up any old destination
if (_destinations != null)
{
for (var i:int = 0; i < _destinations.length; i++)
ObjectPool.give(_destinations[i]);
_destinations.length = 0;
} else
_destinations = new Vector.<SectorVO>;

//add the new destinations
var sector:SectorVO;
for (i = 0; i < sectors.length; i++)
{
sector = ObjectPool.get(SectorVO);
sector.importData(sectors[i]);
sectorIDToSector[sector.id]
```

```
= sector;
_destinations.push(sector);
}
}

public function addPrivateDestinations( sectors:Vector.<SectorData> ):void
{
sectorIDToSector = new Dictionary();
//clear up any old destination
if (_privateDestinations != null)
{
for (var i:int = 0; i < _privateDestinations.length; i++)
ObjectPool.give(_privateDestinations[i]);
_privateDestinations.length = 0;
} else
_privateDestinations = new Vector.<SectorVO>;

//add the new destinations
var sector:SectorVO;
for (i = 0; i < sectors.length; i++)
{
sector = ObjectPool.get(SectorVO);
sector.importData(sectors[i]);
sectorIDToSector[sector.id] = sector;
_privateDestinations.push(sector);
}
}

public function addSectorChangeListener( listener:Function ):void {
_sectorChangeSignal.add(listener); }
public function removeSectorChangeListener( listener:Function ):void {
_sectorChangeSignal.remove(listener); }

public function addSelectedFleetIDChangedListener( listener:Function ):void {
_selectedFleetIDChanged.add(listener); }
public function removeSelectedFleetIDChangedListener( listener:Function ):void {
_selectedFleetIDChanged.remove(listener); }

public function get currentSectorVO():SectorVO { return _sector; }
public function get appearanceSeed():int { return _sector.appearanceSeed; }
public function get depotAsset():String { return _sector.depotAsset; }
public function get destinations():Vector.<SectorVO> { return _destinations; }
public function get privateDestinations():Vector.<SectorVO> { return _privateDestinations; }
public function get height():Number { return _sector.sectorPrototype.getValue("height"); }
public function get neighborhood():int { return _sector.neighborhood; }
public function get numBackgroundSprites():int { return _sector.numBackgroundSprites; }
public function get numPlanetSprites():int { return _sector.numPlanetSprites; }
public function get outpostAsset():String { return _sector.outpostAsset; }
public function get sectorEnum():String { return _sector.sectorEnum; }
public function get sectorFaction():String { return _sector.sectorFaction; }
public
```

```
function get sectorID():String { return _sector.id; }
public function get sectorName():String { return _sector.sectorName; }
public function get sectorPrototype():IPrototype { return _sector.sectorPrototype; }
public function get starbaseAsset():String { return _sector.starbaseAsset; }
public function get starbaseShieldAsset():String { return _sector.starbaseShieldAsset; }
public function get transgateAsset():String { return _sector.transgateAsset; }
public function get width():Number { return _sector.sectorPrototype.getValue("width"); }

public function set focusFleetID( v:String ):void
{
_focusFleetID = v;
_selectedFleetIDChanged.dispatch();
}

public function get focusFleetID():String { return _focusFleetID; }

public function getSectorNameFromID( sectorID:String ):String
{
if (sectorID in sectorIDToSector)
{
var sector:SectorVO = sectorIDToSector[sectorID];
var localization:Localization = Localization.instance;
return localization.getString(sector.sectorName) + ' ' + localization.getString(sector.sectorEnum);
}

return '';
}

public function get igaContextMenuDefaultIndex():int { return _igaContextMenuDefaultIndex; }
public function set igaContextMenuDefaultIndex( v:int ):void { _igaContextMenuDefaultIndex = v;
}

public function get tyrContextMenuDefaultIndex():int { return _tyrContextMenuDefaultIndex; }
public function set tyrContextMenuDefaultIndex( v:int ):void { _tyrContextMenuDefaultIndex = v;
}

public function get sovContextMenuDefaultIndex():int { return _sovContextMenuDefaultIndex; }
public function set sovContextMenuDefaultIndex( v:int ):void { _sovContextMenuDefaultIndex =
v; }

public function get csContextMenuDefaultIndex():int { return _csContextMenuDefaultIndex; }
public function set csContextMenuDefaultIndex( v:int ):void { _csContextMenuDefaultIndex = v; }

}
}

------------------------------------
File 343: igw\com\model\sector\SectorVO.as
package com.model.sector
{
import
```

```
com.model.prototype.IPrototype;
import com.service.server.incoming.data.SectorData;

public class SectorVO
{
private var _appearanceSeed:int;
private var _depotAsset:String;
private var _id:String;
private var _neighborhood:int;
private var _numBackgroundSprites:int;
private var _numPlanetSprites:int;
private var _outpostAsset:String;
private var _sectorEnum:IPrototype;
private var _sectorFaction:String;
private var _sectorName:IPrototype;
private var _sectorPrototype:IPrototype;
private var _starbaseAsset:String;
private var _starbaseShieldAsset:String;
private var _transgateAsset:String;
private var _splitTestCohortPrototype:IPrototype;

public function importData( sector:SectorData ):void
{
_appearanceSeed = sector.appearanceSeed;
_id = sector.id;
_neighborhood = sector.neighborhood;
_sectorEnum = sector.sectorEnum;
_sectorName = sector.sectorName;
_sectorPrototype = sector.prototype;
_splitTestCohortPrototype = sector.splitTestCohortPrototype;

_depotAsset = _sectorPrototype.getValue("depotAsset");
_numBackgroundSprites = _sectorPrototype.getValue("numBackgrounds");
_numPlanetSprites = _sectorPrototype.getValue("numPlanets");
_outpostAsset = _sectorPrototype.getValue("outpostAsset");
_sectorFaction = _sectorPrototype.getValue("factionPrototype");
_starbaseAsset = _sectorPrototype.getValue("baseAsset");
_starbaseShieldAsset = _sectorPrototype.getValue("shieldAsset");
_transgateAsset = _sectorPrototype.getValue("transgateAsset");
}

public function get appearanceSeed():int { return _appearanceSeed; }
public function get depotAsset():String { return _depotAsset; }
public function get height():Number { return _sectorPrototype.getValue("height"); }
public function get id():String { return _id; }
public function get neighborhood():int { return _neighborhood; }
public function get numBackgroundSprites():int { return _numBackgroundSprites; }
public function get numPlanetSprites():int { return _numPlanetSprites; }
public function get outpostAsset():String { return _outpostAsset; }
public function get sectorEnum():String { return _sectorEnum.getValue('nameString'); }
public
```

```
function get sectorEnumPrototype():IPrototype { return _sectorEnum; }
public function get splitTestCohortPrototype():IPrototype { return _splitTestCohortPrototype; }
public function get sectorFaction():String { return _sectorFaction; }
public function get sectorName():String { return _sectorName.getValue('nameString'); }
public function get sectorNamePrototype():IPrototype { return _sectorName; }
public function get sectorPrototype():IPrototype { return _sectorPrototype; }
public function get starbaseAsset():String { return _starbaseAsset; }
public function get starbaseShieldAsset():String { return _starbaseShieldAsset; }
public function get transgateAsset():String { return _transgateAsset; }
public function get width():Number { return _sectorPrototype.getValue("width"); }

public function destroy():void
{
_sectorEnum = null;
_sectorName = null;
_sectorPrototype = null;
}
}
}

------------------------------------
File 344: igw\com\model\starbase\BaseVO.as
package com.model.starbase
{
import com.enum.CurrencyEnum;
import com.enum.StarbaseCategoryEnum;
import com.enum.TypeEnum;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.BaseData;
import com.service.server.incoming.data.BuffData;
import com.service.server.incoming.data.BuildingData;
import com.service.server.incoming.data.ResearchData;
import com.service.server.incoming.data.SectorData;
import com.service.server.incoming.data.TradeRouteData;
import com.util.statcalc.StatCalcUtil;

import flash.utils.Dictionary;
import flash.utils.getTimer;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class BaseVO
{
public var battleServerAddress:String;
public var instancedMissionAddress:String;
public var sectorLocationX:int;
public var sectorLocationY:int;

public
```

```
var onResourcesChange:Signal = new Signal();
public var onResourceDepotChange:Signal = new Signal(BuildingVO);

private var _alloy:uint;
private var _credits:uint;
private var _energy:uint;
private var _synthetic:uint;

private var _bubbleTimeRemaining:Number;
private var _clientTime:int;

private var _buffs:Vector.<BuffVO>;
private var _buffsLookup:Dictionary;
private var _buildings:Vector.<BuildingVO>;
private var _buildingsLookup:Dictionary;
private var _research:Vector.<ResearchVO>;
private var _researchLookup:Dictionary;
private var _tradeRouteID:int = 0;
private var _tradeRoutes:Vector.<TradeRouteVO>;
private var _tradeRouteLookup:Dictionary;

private var _disableReqs:Number = 0;
private var _grid:StarbaseGrid;
private var _id:String;

private var _baseResourceIncome:uint;
private var _baseCreditIncome:uint;
private var _baseResourcePurchaseScale:Number;
private var _baseCreditPurchaseScale:Number;
private var _sector:SectorData;
private var _tradeRouteResourceIncome:uint;
private var _tradeRouteCreditIncome:uint;
private var _derivedStatsDirty:Boolean = true;
private var _maxResources:int;
private var _maxCredits:int;

public function init( baseData:BaseData ):void
{
_id = baseData.id;
_buffs = new Vector.<BuffVO>;
_buffsLookup = new Dictionary();
_buildings = new Vector.<BuildingVO>;
_buildingsLookup = new Dictionary();
_grid = new StarbaseGrid();
_research = new Vector.<ResearchVO>;
_researchLookup = new Dictionary();
}

internal function importData( baseData:BaseData ):void
{
_clientTime
```

```
= getTimer();
_alloy = baseData.alloy;
_credits = baseData.credits;
_energy = baseData.energy;
_synthetic = baseData.synthetic;
_bubbleTimeRemaining = baseData.bubbleTimeRemaining;

_sector = baseData.sector;
sectorLocationX = baseData.sectorLocationX;
sectorLocationY = baseData.sectorLocationY;
onResourcesChange.dispatch();
}


//==========================================================================
//**************************************************************************************************
// BUILDINGS
//**************************************************************************************************

//==========================================================================

internal function importBuildingData( buildingData:BuildingData ):void
{
if (!_buildingsLookup[buildingData.id])
{
var buildingVO:BuildingVO = ObjectPool.get(BuildingVO);
buildingVO.init(buildingData);
addBuilding(buildingVO);
} else
_buildingsLookup[buildingData.id].importData(buildingData);
_derivedStatsDirty = true;
}

internal function addBuilding( vo:BuildingVO ):void
{
_buildings.push(vo);
_buildingsLookup[vo.id] = vo;
_grid.addToGrid(vo, true);
}

internal function updateBuildingID( oldID:String, newID:String ):void
{
//ensure that the server didn't beat us to it
if (_buildingsLookup[newID] != null)
{
_buildingsLookup[oldID] = null;
delete _buildingsLookup[oldID];
} else
{
var buildingVO:BuildingVO = _buildingsLookup[oldID];
_buildingsLookup[oldID]
```

```
= null;
delete _buildingsLookup[oldID];
buildingVO.forceSetID(newID);
_buildingsLookup[newID] = buildingVO;
}
}

public function getBuildingCount( buildingClass:String ):int
{
var count:int = 0;
var countStarbaseStructures:Boolean = (buildingClass == TypeEnum.STARBASE_ARM ||
buildingClass == TypeEnum.STARBASE_PLATFORMA ||
buildingClass == TypeEnum.STARBASE_PLATFORMB);
for (var i:int = 0; i < _buildings.length; i++)
{
if (countStarbaseStructures)
{
if (_buildings[i].getValue('category') == StarbaseCategoryEnum.STARBASE_STRUCTURE)
count += (_buildings[i].sizeX / 5) * (_buildings[i].sizeY / 5);
} else if (_buildings[i].itemClass == buildingClass)
count++;
}
return count;
}

public function getBuildingMaxCount( buildingClass:String ):int
{
switch (buildingClass)
{
case TypeEnum.POINT_DEFENSE_PLATFORM:
return StatCalcUtil.baseStatCalc("MaxTurret");
case TypeEnum.PYLON:
return StatCalcUtil.baseStatCalc("MaxPylon");
case TypeEnum.SHIELD_GENERATOR:
return StatCalcUtil.baseStatCalc("MaxShield");
case TypeEnum.RESOURCE_DEPOT:
return StatCalcUtil.baseStatCalc("MaxDepot");
case TypeEnum.STARBASE_ARM:
case TypeEnum.STARBASE_PLATFORMA:
case TypeEnum.STARBASE_PLATFORMB:
return StatCalcUtil.baseStatCalc("MaxPlatform");
case TypeEnum.STARBASE_WALL:
return StatCalcUtil.baseStatCalc("MaxWall");
}
if(CONFIG::DEBUG == true)
{
switch (buildingClass)
{
case TypeEnum.COMMAND_CENTER:
return StatCalcUtil.baseStatCalc("MaxCommand");
case
```

```
TypeEnum.ADVANCED_TECH:
case TypeEnum.DEFENSE_DESIGN:
case TypeEnum.SHIPYARD:
case TypeEnum.WEAPONS_FACILITY:
return StatCalcUtil.baseStatCalc("MaxResearch");
case TypeEnum.SURVEILLANCE:
return StatCalcUtil.baseStatCalc("MaxSurveillance");
case TypeEnum.DOCK:
return StatCalcUtil.baseStatCalc("MaxDock");
case TypeEnum.CONSTRUCTION_BAY:
return StatCalcUtil.baseStatCalc("MaxShipyard");
case TypeEnum.REACTOR_STATION:
return StatCalcUtil.baseStatCalc("MaxReactor");
}
}
return 1;
}

internal function getBuildingByID( id:String ):BuildingVO
{
if (_buildingsLookup[id])
return _buildingsLookup[id];
return null;
}

internal function getBuildingByClass( buildingClass:String, highestLevel:Boolean = false
):BuildingVO
{
var vo:BuildingVO;
for (var i:int = 0; i < _buildings.length; i++)
{
if (_buildings[i].itemClass == buildingClass)
{
if (!highestLevel)
return _buildings[i];
else if (!vo || vo.level < _buildings[i].level)
vo = _buildings[i];
}
}
return vo;
}

internal function removeBuilding( buildingVO:BuildingVO ):void
{
var index:int = _buildings.indexOf(buildingVO);
_buildings.splice(index, 1);
_grid.removeFromGrid(buildingVO);
//remove from the lookup
delete _buildingsLookup[buildingVO.id];
_buildingsLookup[buildingVO.id] = null;
_derivedStatsDirty
```

```
= true;
}

internal function get buildings():Vector.<BuildingVO> { return _buildings; }


//==========================================================================================
//******************************************************************************************
// RESEARCH
//******************************************************************************************

internal function importResearchData( researchData:ResearchData ):void
{
if (!_researchLookup[researchData.id])
{
var researchVO:ResearchVO = ObjectPool.get(ResearchVO);
researchVO.init(researchData.id);
_research.push(researchVO);
_researchLookup[researchVO.id] = researchVO;
}
_researchLookup[researchData.id].importData(researchData);
_derivedStatsDirty = true;
}

internal function getResearchByID( id:String ):ResearchVO
{
if (_researchLookup[id])
return _researchLookup[id];
return null;
}

internal function removeResearch( researchVO:ResearchVO ):void
{
var index:int = _research.indexOf(researchVO);
_research.splice(index, 1);
//remove from the lookup
ObjectPool.give(_researchLookup[researchVO.id]);
delete _researchLookup[researchVO.id];
_researchLookup[researchVO.id] = null;
_derivedStatsDirty = true;
}

internal function get research():Vector.<ResearchVO> { return _research; }


//==========================================================================================
//******************************************************************************************
// BUFFS
//******************************************************************************************

//==========================================================================================
```

```
internal function importBuffData( buffData:BuffData ):void
{
var added:Boolean = false;
if (!_buffsLookup[buffData.id])
{
var buffVO:BuffVO = ObjectPool.get(BuffVO);
_buffsLookup[buffData.id] = buffVO;
_buffs.push(buffVO);
added = true;
}
_buffsLookup[buffData.id].importData(buffData);
if (added)
{
_derivedStatsDirty = true;
_disableReqs = StatCalcUtil.baseStatCalc("disableReqs");
}
}

internal function getBuffByID( id:String ):BuffVO
{
if (_buffsLookup[id])
return _buffsLookup[id];
return null;
}

internal function getBuffByType( type:String ):BuffVO
{
for (var i:int = 0; i < _buffs.length; i++)
{
if (_buffs[i].buffType == type)
return _buffs[i];
}
return null;
}

internal function updateBuffID( oldID:String, newID:String ):void
{
//ensure that the server didn't beat us
if (_buffsLookup[newID] != null)
{
for (var i:int = 0; i < _buffs.length; i++)
{
if (_buffs[i].id == oldID)
{
_buffs.splice(i, 1);
break;
}
}

_buffsLookup[oldID]
```

```
= null;
delete _buffsLookup[oldID];
} else
{
var buffVO:BuffVO = _buffsLookup[oldID];
_buffsLookup[oldID] = null;
delete _buffsLookup[oldID];
buffVO.forceSetID(newID);
_buffsLookup[newID] = buffVO;
}
}

internal function removeBuff( buffVO:BuffVO ):void
{
var index:int = _buffs.indexOf(buffVO);
_buffs.splice(index, 1);
//remove from the lookup
ObjectPool.give(_buffsLookup[buffVO.id]);
delete _buffsLookup[buffVO.id];
_buffsLookup[buffVO.id] = null;
_disableReqs = StatCalcUtil.baseStatCalc("disableReqs");
_derivedStatsDirty = true;
}

public function get buffs():Vector.<BuffVO> { return _buffs; }
public function get reqsDisabled():Number { return _disableReqs; }


//==========================================================================
//**************************************************************************************************
// TRADE ROUTES
//**************************************************************************************************

//==========================================================================

internal function initTradeRoutes():void
{
_tradeRoutes = new Vector.<TradeRouteVO>;
_tradeRouteLookup = new Dictionary();
setUpDefaultTradeRoutes();
}

internal function importTradeRouteData( tradeRouteData:TradeRouteData ):void
{
if (!_tradeRouteLookup[tradeRouteData.id])
{
//can only have one trade route per faction
//if we're getting a traderoute from the server that does not match the id but matches the faction
then update the id
if (_tradeRouteLookup[tradeRouteData.corporation])
{
```

```
updateTradeRouteID(_tradeRouteLookup[tradeRouteData.corporation].id, tradeRouteData.id);
} else
{
var tradeRouteVO:TradeRouteVO = ObjectPool.get(TradeRouteVO);
tradeRouteVO.init(tradeRouteData.id);
_tradeRoutes.push(tradeRouteVO);
_tradeRouteLookup[tradeRouteVO.id] = tradeRouteVO;
_tradeRouteLookup[tradeRouteData.corporation] = tradeRouteVO;
}
}
_tradeRouteLookup[tradeRouteData.id].importData(tradeRouteData);
_derivedStatsDirty = true;
}

internal function updateTradeRouteID( oldID:String, newID:String ):void
{
//ensure that the server didn't beat us to it
if (_tradeRouteLookup[newID] != null)
{
_tradeRouteLookup[oldID] = null;
delete _tradeRouteLookup[oldID];
} else
{
var tradeRouteVO:TradeRouteVO = _tradeRouteLookup[oldID];
_tradeRouteLookup[oldID] = null;
delete _tradeRouteLookup[oldID];
tradeRouteVO.forceSetID(newID);
_tradeRouteLookup[newID] = tradeRouteVO;
}
}

internal function getTradeRouteByID( id:String ):TradeRouteVO
{
if (_tradeRouteLookup[id])
return _tradeRouteLookup[id];
return null;
}

internal function getTradeRouteByCorporation( corportation:String ):TradeRouteVO
{
if (_tradeRouteLookup[corportation])
return _tradeRouteLookup[corportation];
return null;
}

internal function removeTradeRoute( tradeRouteVO:TradeRouteVO ):void
{
var index:int = _tradeRoutes.indexOf(tradeRouteVO);
_tradeRoutes.splice(index, 1);
//remove
```

```
from the lookup
ObjectPool.give(_tradeRouteLookup[tradeRouteVO.id]);
_tradeRouteLookup[tradeRouteVO.id] = null;
delete _tradeRouteLookup[tradeRouteVO.id];
_tradeRouteLookup[tradeRouteVO.corporation] = null;
delete _tradeRouteLookup[tradeRouteVO.corporation];
_derivedStatsDirty = true;
}

private function setUpDefaultTradeRoutes():void
{
var factions:Vector.<IPrototype> = PrototypeModel.instance.getFactionPrototypes();
var currentFaction:IPrototype;
var tradeRouteData:TradeRouteData;
for (var i:uint = 0; i < factions.length; ++i)
{
currentFaction = factions[i];
if (currentFaction.getValue('contractGroup') != "")
{
tradeRouteData = ObjectPool.get(TradeRouteData);
tradeRouteData.baseID = _id;
tradeRouteData.id = tradeRouteID;
tradeRouteData.factionPrototype = factions[i];
tradeRouteData.reputation = 0;
importTradeRouteData(tradeRouteData);
}
}
}

internal function get tradeRoutes():Vector.<TradeRouteVO> { return _tradeRoutes; }
private function get tradeRouteID():String { ++_tradeRouteID; return CurrentUser.name +
'.clientside_tradeRoute.' + String(_tradeRouteID); }


//=============================================================================
//*****************************************************************************
// RESOURCE MANAGEMENT
//*****************************************************************************

//=============================================================================

public function updateResources():void
{
if (_derivedStatsDirty)
{
calcDerivedStats();
_derivedStatsDirty = false;
}

// clamp resources to their maximums
var
```

```
maxCredit:int = maxCredits;
var maxResource:int = maxResources;
_alloy = Math.min(maxResource, _alloy);
_credits = Math.min(maxCredit, _credits);
_energy = Math.min(maxResource, _energy);
_synthetic = Math.min(maxResource, _synthetic);

onResourcesChange.dispatch();
}

public function deposit( amount:uint, type:String ):void
{
switch (type)
{
case CurrencyEnum.ALLOY:
_alloy += amount;
break;
case CurrencyEnum.CREDIT:
_credits += amount;
break;
case CurrencyEnum.ENERGY:
_energy += amount;
break;
case CurrencyEnum.SYNTHETIC:
_synthetic += amount;
break;
}

onResourcesChange.dispatch();
}

public function withdraw( amount:uint, type:String ):Boolean
{
switch (type)
{
case CurrencyEnum.ALLOY:
if (_alloy < amount)
return false;
_alloy -= amount;
break;
case CurrencyEnum.CREDIT:
if (_credits < amount)
return false;
_credits -= amount;
break;
case CurrencyEnum.ENERGY:
if (_energy < amount)
return false;
_energy -= amount;
break;
case
```

```
CurrencyEnum.SYNTHETIC:
if (_synthetic < amount)
return false;
_synthetic -= amount;
break;
}
onResourcesChange.dispatch();
return true;
}

public function calcDerivedStats():void
{
_baseResourceIncome = StatCalcUtil.baseStatCalc("ExpectedResourceIncome");
_baseCreditIncome = StatCalcUtil.baseStatCalc("ExpectedCreditIncome");

_baseResourcePurchaseScale = StatCalcUtil.baseStatCalc("resourceBuyScale");
_baseCreditPurchaseScale = StatCalcUtil.baseStatCalc("creditBuyScale");

_tradeRouteResourceIncome = StatCalcUtil.baseStatCalc("ResIncome");
_tradeRouteCreditIncome = StatCalcUtil.baseStatCalc("CredIncome");

var newResourceCap:int = StatCalcUtil.baseStatCalc("ResourceCap");
var newCreditsCap:int = StatCalcUtil.baseStatCalc("CreditCap");

if (newResourceCap != _maxResources || newCreditsCap != _maxCredits)
{
_maxCredits = newCreditsCap;
_maxResources = newResourceCap;
onResourcesChange.dispatch();
}
}

public function set dirty( v:Boolean ):void { _derivedStatsDirty = true; }

public function get grid():StarbaseGrid { return _grid; }

public function get alloy():uint { return _alloy; }
public function get credits():uint { return _credits; }
public function get energy():uint { return _energy; }
public function get synthetic():uint { return _synthetic; }

public function get bubbleTimeRemaining():Number
{
var temp:Number = _bubbleTimeRemaining - (getTimer() - _clientTime);
if (temp < 0)
temp = 0;
return temp;
}

public function get baseResourceIncome():uint { return _baseResourceIncome; }
public
```

```
function get baseCreditIncome():uint { return _baseCreditIncome; }

public function get baseResourcePurchaseScale():Number { return
_baseResourcePurchaseScale; }
public function get baseCreditPurchaseScale():Number { return _baseCreditPurchaseScale; }

public function get tradeRouteResourceIncome():uint { return _tradeRouteResourceIncome; }
public function get tradeRouteCreditIncome():uint { return _tradeRouteCreditIncome; }

public function get id():String { return _id; }

public function get maxPower():int { return StatCalcUtil.baseStatCalc("MaxPower"); }
public function get maxResources():int { return _maxResources; }
public function get maxCredits():int { return _maxCredits; }

public function get sector():SectorData { return _sector; }
public function get sectorID():String { return _sector.id; }

public function destroy():void
{

}
}
}



-----------------------------------
File 345: igw\com\model\starbase\BuffVO.as
package com.model.starbase
{
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.BuffData;

import flash.utils.getTimer;

public class BuffVO implements IPrototype
{
public var baseID:String;
public var playerOwnerID:String;
public var prototypeVO:IPrototype;
public var began:Number;
public var ends:Number;
public var timeRemaining:Number;

private var _timeRemainingMS:Number;
private var _clientTime:Number;
private var _id:String;

internal function importData( buffData:BuffData ):void
{
```

```
_id = buffData.id;
prototypeVO = buffData.prototype;
baseID = buffData.baseID;
playerOwnerID = buffData.playerOwnerID;
began = buffData.began;
ends = buffData.ends;
timeRemaining = buffData.timeRemaining;

_clientTime = getTimer();
}

internal function forceSetID( v:String ):void { _id = v; }

public function getUnsafeValue( key:String ):* { return prototypeVO.getUnsafeValue(key); }
public function getValue( key:String ):* { return prototypeVO.getValue(key); }

public function get asset():String { return prototypeVO.asset; }
public function get uiAsset():String { return prototypeVO.uiAsset; }

public function get name():String { return prototypeVO.name; }
public function get id():String { return _id }
public function get itemClass():String { return prototypeVO.itemClass; }
public function get buildTimeSeconds():uint { return prototypeVO.buildTimeSeconds; }

public function get timeRemainingMS():Number
{
_timeRemainingMS = timeRemaining - (getTimer() - _clientTime);
if (_timeRemainingMS < 0)
_timeRemainingMS = 0;
return _timeRemainingMS;
}

public function get buffType():String { return getValue('buffType'); }

public function get alloyCost():int { return 0; }
public function get creditsCost():int { return 0; }
public function get energyCost():int { return 0; }
public function get syntheticCost():int { return 0; }
}
}

------------------------------------
File 346: igw\com\model\starbase\BuildingVO.as
package com.model.starbase
{
import com.enum.TypeEnum;
import com.enum.server.StarbaseBuildStateEnum;
import com.game.entity.components.shared.Detail;
import com.model.fleet.EmptySlotVO;
import
```

```
com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.data.BuildingData;
import com.util.CommonFunctionUtil;
import com.util.statcalc.StatCalcUtil;

import flash.utils.Dictionary;

import org.ash.core.Entity;
import org.osflash.signals.Signal;

public class BuildingVO implements IPrototype
{
public static const EMPTY_SLOT:IPrototype = new EmptySlotVO();

public var baseID:String;
public var damaged:Boolean = false;
public var destroyed:Boolean = false;
public var baseX:Number;
public var baseY:Number;
public var buildState:int;
public var modules:Dictionary;
public var percentFilled:Number; //used by resource depots to show how full they are
public var refitModules:Dictionary;

private var _refitBuildCreditCost:int = 0;
private var _refitBuildAlloyCost:int = 0;
private var _refitBuildEnergyCost:int = 0;
private var _refitBuildSyntheticCost:int = 0;
private var _refitBuildTime:int = 0;

private var _currentHealth:Number = 1;
private var _buildingDps:int = 0;
private var _built:Boolean;
private var _forceShielding:int = 0;
private var _explosiveShielding:int = 0;
private var _energyShielding:int = 0;

private var _slots:Array;

private var _id:String;
private var _onHealthChange:Signal = new Signal(Number, Number);
private var _playerOwnerID:String;
private var _prototype:IPrototype;
private var _shieldRadius:int; //used by shield generators
private var _type:String;

public function BuildingVO()
{
modules
```

```
= new Dictionary();
refitModules = new Dictionary();
}

internal function init( buildingData:BuildingData ):void
{
baseID = buildingData.baseID;
_id = buildingData.id;
_built = _id.indexOf('clientside') == -1;
percentFilled = 0;
_playerOwnerID = buildingData.playerOwnerID;
_shieldRadius = -1;
_type = buildingData.type;
importData(buildingData);
calculateCosts();
}

internal function importData( buildingData:BuildingData ):void
{
baseID = buildingData.baseID;
baseX = buildingData.baseX;
baseY = buildingData.baseY;
buildState = buildingData.buildState;
currentHealth = buildingData.currentHealth;
modules = buildingData.modules ? buildingData.modules : modules;
prototype = buildingData.prototype;
refitModules = buildingData.refitModules ? buildingData.refitModules : refitModules;

if (prototype)
_slots = prototype.getValue('slots');

}

public function popuplateFromEntity( entity:Entity ):void
{
_id = entity.id;
_playerOwnerID = CurrentUser.id;
_type = Detail(entity.get(Detail)).type;
}

public function equipModule( vo:IPrototype, slot:String ):void
{
modules[slot] = vo;
calculateCosts();
}

public function equipRefitModule( vo:IPrototype, slot:String ):void
{
refitModules[slot] = vo;
calculateCosts();
}
```

```
public function calculateCosts():void
{
var vo:IPrototype;
var refitVO:IPrototype;
_buildingDps = 0;
_forceShielding = 0;
_explosiveShielding = 0;
_energyShielding = 0;

_refitBuildCreditCost = 0;
_refitBuildAlloyCost = 0;
_refitBuildEnergyCost = 0;
_refitBuildSyntheticCost = 0;
_refitBuildTime = 0;

if (_slots)
{
var len:uint = _slots.length;
var slot:String;
for (var i:uint = 0; i < len; ++i)
{
slot = _slots[i];

vo = modules[slot];
refitVO = refitModules[slot];
vo = (refitVO != null) ? (refitVO != EMPTY_SLOT) ? refitVO : null : vo;
if (vo)
{
_refitBuildCreditCost += vo.creditsCost;
_refitBuildAlloyCost += vo.alloyCost;
_refitBuildEnergyCost += vo.energyCost;
_refitBuildSyntheticCost += vo.syntheticCost;
_refitBuildTime += vo.buildTimeSeconds;

_buildingDps += calcSingleDps(vo, slot);
_forceShielding = calcShielding(vo, slot, 1);
_explosiveShielding = calcShielding(vo, slot, 2);
_energyShielding = calcShielding(vo, slot, 3);
}
}
}

_refitBuildCreditCost *= PrototypeModel.BUILDING_RESOURCE_COST_SCALAR;
_refitBuildAlloyCost *= PrototypeModel.BUILDING_RESOURCE_COST_SCALAR;
_refitBuildEnergyCost *= PrototypeModel.BUILDING_RESOURCE_COST_SCALAR;
_refitBuildSyntheticCost *= PrototypeModel.BUILDING_RESOURCE_COST_SCALAR;
_refitBuildTime *= PrototypeModel.BUILDING_BUILD_TIME_SCALAR;
}

private
```

```
function calcSingleDps( mod:IPrototype, slot:String ):Number
{
var slotType:String = mod.getValue('slotType');
if (!(slotType == "Weapon" || slotType == "Spinal" || slotType == "Arc" || slotType == "Drone" ||
slotType == "BaseTurret"))
return 0;

var fireTime:Number = StatCalcUtil.entityStatCalc(this, 'fireTime', 0, mod, slot);
var burstSize:Number = StatCalcUtil.entityStatCalc(this, 'burstSize', 0, mod, slot);
var reloadTime:Number = StatCalcUtil.entityStatCalc(this, 'reloadTime', 0, mod, slot);
var chargeTime:Number = StatCalcUtil.entityStatCalc(this, 'chargeTime', 0, mod, slot);
var volleySize:Number = StatCalcUtil.entityStatCalc(this, 'volleySize', 0, mod, slot);
var duration:Number = StatCalcUtil.entityStatCalc(this, 'duration', 0, mod, slot);
var tickRate:Number = StatCalcUtil.entityStatCalc(this, 'tickRate', 0, mod, slot);
var damageTime:Number = StatCalcUtil.entityStatCalc(this, 'damageTime', 0, mod, slot);
var maxDrones:Number = StatCalcUtil.entityStatCalc(this, 'maxDrones', 0, mod, slot);
var damage:Number = StatCalcUtil.entityStatCalc(this, 'damage', 0, mod, slot);

// Do the appropriate calculation based on the attack type
var type:int = mod.getValue('attackMethod');
var totalDamage:Number = 0;
var totalPeriod:Number = 0;
if (type == 1 || type == 2 || type == 3)
{
// Beams and Projectiles
totalDamage = damage * Math.max(burstSize, 1) * Math.max(volleySize, 1);
totalPeriod = (fireTime * burstSize) + reloadTime + chargeTime;
} else if (type == 4)
{
// Areas
totalDamage = damage * Math.max(burstSize, 1) * Math.max(volleySize, 1) * Math.max(duration
/ tickRate, 1);
totalPeriod = (fireTime * burstSize) + reloadTime + chargeTime + duration;
} else if (type == 5)
{
// Drones
totalDamage = damage * Math.max(maxDrones, 1);
totalPeriod = damageTime;
}

var DPS:Number = totalDamage / totalPeriod;
return DPS;
}

private function calcShielding( mod:IPrototype, slot:String, type:Number ):Number
{
var typeShielding:Number = 0;
switch (type)
{
case 1: // Force
typeShielding
```

```
= StatCalcUtil.entityStatCalc(this, 'forceShielding', 0, mod, slot);
break;
case 2: // Explosive
typeShielding = StatCalcUtil.entityStatCalc(this, 'explosiveShielding', 0, mod, slot);
break;
case 3: // Energy
typeShielding = StatCalcUtil.entityStatCalc(this, 'energyShielding', 0, mod, slot);
break;
}

var shielding:Number = StatCalcUtil.entityStatCalc(this, 'shielding', 0, mod, slot);

return (typeShielding + shielding);
}

internal function forceSetID( v:String ):void { _id = v; }

public function getUnsafeValue( key:String ):* { return _prototype.getUnsafeValue(key); }
public function getValue( key:String ):* { return _prototype.getValue(key); }

public function get asset():String { return _prototype.asset; }
public function get uiAsset():String { return _prototype.uiAsset; }

public function get name():String { return _prototype.name; }
public function get itemClass():String { return _prototype.itemClass; }
public function get buildTimeSeconds():uint { return (buildState ==
StarbaseBuildStateEnum.DONE) ? _refitBuildTime : (_prototype.buildTimeSeconds *
PrototypeModel.BUILDING_BUILD_TIME_SCALAR); }
public function get refitBuildTimeSeconds():uint { return _refitBuildTime; }

public function get alloyCost():int { return (buildState == StarbaseBuildStateEnum.DONE) ?
_refitBuildAlloyCost : (_prototype.alloyCost *
PrototypeModel.BUILDING_RESOURCE_COST_SCALAR); }
public function get creditsCost():int { return (buildState == StarbaseBuildStateEnum.DONE) ?
_refitBuildCreditCost : (_prototype.creditsCost *
PrototypeModel.BUILDING_RESOURCE_COST_SCALAR); }
public function get energyCost():int { return (buildState == StarbaseBuildStateEnum.DONE) ?
_refitBuildEnergyCost : (_prototype.energyCost *
PrototypeModel.BUILDING_RESOURCE_COST_SCALAR); }
public function get syntheticCost():int { return (buildState == StarbaseBuildStateEnum.DONE) ?
_refitBuildSyntheticCost : (_prototype.syntheticCost *
PrototypeModel.BUILDING_RESOURCE_COST_SCALAR); }

public function get constructionCategory():String { return
_prototype.getValue('constructionCategory'); }

public function get currentHealth():Number { return _currentHealth; }
public function set currentHealth( v:Number ):void
{
var change:Number = (_currentHealth - v);
_currentHealth
```

```
= v;
_onHealthChange.dispatch(_currentHealth, change);
}

public function get level():int { return _prototype.getValue('level'); }

public function get id():String { return _id; }

public function get playerOwnerID():String { return _playerOwnerID; }

public function get prototype():IPrototype { return _prototype; }
public function set prototype( v:IPrototype ):void
{
_prototype = v;
if (_shieldRadius != -1)
{
if (_prototype.itemClass == TypeEnum.SHIELD_GENERATOR)
_shieldRadius = StatCalcUtil.baseStatCalc('radius', 0.0, _id);
else if (_prototype.itemClass == TypeEnum.PYLON)
_shieldRadius = StatCalcUtil.baseStatCalc('maxWallLength', 0.0, _id);
}
}

public function get shieldRadius():int
{
if (_shieldRadius == -1)
{
if (_prototype.itemClass == TypeEnum.SHIELD_GENERATOR)
_shieldRadius = StatCalcUtil.baseStatCalc('radius', 0.0, _id);
else if (_prototype.itemClass == TypeEnum.PYLON)
_shieldRadius = StatCalcUtil.baseStatCalc('maxWallLength', 0.0, _id);
}
return _shieldRadius;
}

public function get sizeX():int { return _prototype.getValue('sizeX'); }
public function get sizeY():int { return _prototype.getValue('sizeY'); }

public function get type():String { return _type; }

public function addHealthListener( callback:Function ):void { _onHealthChange.add(callback); }
public function removeHealthListener( callback:Function ):void {
_onHealthChange.remove(callback); }

public function get buildingDps():int { return _buildingDps; }

public function get built():Boolean
{
if (_built)
return true;
else
```

```
if (_id && _id.indexOf('clientside') == -1)
_built = true;
return _built;
}

public function get forceShielding():int { return _forceShielding; }

public function get explosiveShielding():int { return _explosiveShielding; }

public function get energyShielding():int { return _energyShielding; }


}
}
```

File 347: igw\com\model\starbase\ResearchVO.as

```
package com.model.starbase
{
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.ResearchData;

public class ResearchVO implements IPrototype
{
public var baseID:String;
public var playerOwnerID:String;
public var prototype:IPrototype;

private var _id:String = '';

internal function init( id:String ):void
{
_id = id;
}

public function importData( researchData:ResearchData ):void
{
baseID = researchData.baseID;
playerOwnerID = researchData.playerOwnerID;
prototype = researchData.prototype;
}

internal function forceSetID( v:String ):void { _id = v; }

public function getUnsafeValue( key:String ):* { return prototype.getUnsafeValue(key); }
public function getValue( key:String ):* { return prototype.getValue(key); }

public function get id():String { return _id; }

public function get name():String { return prototype.name; }
public
```

```
function get itemClass():String { return prototype.itemClass; }
public function get level():int { return prototype.getValue('level'); }
public function get buildTimeSeconds():uint { return prototype.buildTimeSeconds; }
public function get requiredBuildingClass():String { return
prototype.getValue('requiredBuildingClass'); }

public function get asset():String { return prototype.asset; }
public function get uiAsset():String { return prototype.uiAsset; }

public function get alloyCost():int { return prototype.alloyCost; }
public function get creditsCost():int { return prototype.creditsCost; }
public function get energyCost():int { return prototype.energyCost; }
public function get syntheticCost():int { return prototype.syntheticCost; }
}
}

-------------------------------------
File 348: igw\com\model\starbase\StarbaseGrid.as
package com.model.starbase
{
import com.enum.StarbaseConstructionEnum;
import com.enum.TypeEnum;
import com.game.entity.factory.IStarbaseFactory;

import flash.geom.Point;
import flash.geom.Rectangle;
import flash.utils.Dictionary;

import org.ash.core.Game;

public class StarbaseGrid
{
public static const ALL_CLEAR:int = 0;
public static const GRID_OCCUPIED:int = 1;

public static const PLAY_SPACE_WIDTH:int = 5000;
public static const PLAY_SPACE_NOBUILD_WIDTH:int = 540;
public static const PLAY_SPACE_HEIGHT:int = 5000;
public static const PLAY_SPACE_NOBUILD_HEIGHT:int = 540;

private const OPEN_SPACE:int = 0;
private const PLATFORM:int = 1;
private const BUILDING:int = 2;
private const WALL:int = 3;

private const GRID_SIZE:Number = 18;
private const CELL_SIZE_SMALL_HALF:int = GRID_SIZE * .5;
private const CELL_SIZE:int = 90; //90 is size of a single platform

private var COLUMNS_SMALL:int;
private
```

```
var CONVERSION_SMALL:Number;
private var COLUMNS:int;
private var CONVERSION:Number;
private var REAL_SIZE:int;

private const REAL_OFFSET:Point = new Point();
private const ISO_SCALE:Number = 0.5;
private const ISO_ROTATION:Number = 45 * 0.0174532925;
private const PIVOT:Point = new Point();

private var _columnSpan:Point;
private var _position:Point;
private var _rect:Rectangle;
private var _rowSpan:Point;
private var _space:Dictionary;
private var _testSpan:Point;

public function StarbaseGrid()
{
_space = new Dictionary();
_columnSpan = new Point();
_position = new Point();
_rect = new Rectangle();
_rowSpan = new Point();
_testSpan = new Point();

CONVERSION = 1 / CELL_SIZE;
CONVERSION_SMALL = 1 / GRID_SIZE;

initIso(PLAY_SPACE_WIDTH, PLAY_SPACE_HEIGHT);
COLUMNS_SMALL = Math.ceil(REAL_SIZE / GRID_SIZE);
COLUMNS = Math.ceil(REAL_SIZE / CELL_SIZE);

placeBase();
}

public function addToGrid( vo:BuildingVO, ignoreChecks:Boolean = false ):int
{
var target:int = (vo.constructionCategory == StarbaseConstructionEnum.PLATFORM ||
vo.itemClass == TypeEnum.PYLON) ? OPEN_SPACE : PLATFORM;
var result:int = ALL_CLEAR;
if (ignoreChecks)
addOrRemoveFromGrid(vo, target, false, false, ignoreChecks);
else
{
result = confirmAddToGrid(vo);
if (result == ALL_CLEAR)
addOrRemoveFromGrid(vo, target, false, false);
}
return result;
}
```

```
public function showGrid( game:Game, factory:IStarbaseFactory ):void
{
var p:Point = new Point();
for (var i:int = 0; i < COLUMNS_SMALL; i += 5)
{
for (var j:int = 0; j < COLUMNS_SMALL; j += 5)
{
p.setTo(i, j);
convertGridToIso(p);
factory.createGridSquare(TypeEnum.ISO_1x1_SOVEREIGNTY, p.x, p.y);
}
}
}

public function confirmAddToGrid( vo:BuildingVO ):int
{
var target:int = (vo.constructionCategory == StarbaseConstructionEnum.PLATFORM ||
vo.itemClass == TypeEnum.PYLON) ? OPEN_SPACE : PLATFORM;
return addOrRemoveFromGrid(vo, target, true);
}

public function removeFromGrid( vo:BuildingVO ):void
{
addOrRemoveFromGrid(vo, (vo.constructionCategory ==
StarbaseConstructionEnum.PLATFORM || vo.itemClass == TypeEnum.PYLON) ?
OPEN_SPACE : PLATFORM, false, true);
}

public function snapToGrid( position:Point, vo:BuildingVO, snapToSmallCells:Boolean = true
):void
{
if (!snapToSmallCells)
{
convertPointFromIsometric(position);
getKey(getHash(position.x, position.y), position);
//center on the grid position if an odd size
if (vo.sizeX % 10 != 0)
position.x += 45;
if (vo.sizeY % 10 != 0)
position.y += 45;
convertPointToIsometric(position);
convertBuildingIsoToGrid(position, vo);
} else
{
convertPointFromIsometric(position);
getSmallKey(getSmallHash(position.x, position.y), position);
//center on the grid position if an odd size
if (vo.sizeX % 10 != 0)
{
```

```
position.x += CELL_SIZE_SMALL_HALF;
position.y += CELL_SIZE_SMALL_HALF;
}
convertPointToIsometric(position);
convertBuildingIsoToGrid(position, vo);
}
}

public function getHash( keyX:Number, keyY:Number ):int
{
return (keyX * CONVERSION | 0) + (keyY * CONVERSION | 0) * COLUMNS;
}

public function getKey( hash:int, point:Point ):void
{
point.x = (hash % COLUMNS) * CELL_SIZE;
point.y = (hash / COLUMNS | 0) * CELL_SIZE;
}

public function getSmallHash( keyX:Number, keyY:Number ):int
{
return (keyX * CONVERSION_SMALL | 0) + (keyY * CONVERSION_SMALL | 0) *
COLUMNS_SMALL;
}

public function getSmallKey( hash:int, point:Point ):void
{
point.x = (hash % COLUMNS_SMALL) * GRID_SIZE;
point.y = (hash / COLUMNS_SMALL | 0) * GRID_SIZE;
}

//We only want the player to move a platform if a building is not sitting on top of it
//and if moving it won't create a narrow channel
public function canMovePlatform( vo:BuildingVO ):Boolean
{
//check for buildings
_position.setTo(vo.baseX * GRID_SIZE, vo.baseY * GRID_SIZE);
var hashTL:int = getSmallHash(_position.x, _position.y);
var hashBR:int = getSmallHash(_position.x + ((vo.sizeX - 1) * GRID_SIZE), _position.y +
((vo.sizeY - 1) * GRID_SIZE));
var len:int = hashBR % COLUMNS_SMALL;
var i:int = hashTL;
while (i <= hashBR)
{
if (_space[i] == BUILDING)
return false;
if (i % COLUMNS_SMALL == len)
{
hashTL += COLUMNS_SMALL;
i
```

```
= hashTL;
} else
i++;
}

return true;
}

private function addOrRemoveFromGrid( vo:BuildingVO, target:int, checking:Boolean = false,
remove:Boolean = false, ignoreChecks:Boolean = false ):int
{
//check within buildable bounds
convertBuildingGridToIso(_position, vo);
if (_position.x < PLAY_SPACE_NOBUILD_WIDTH || _position.x > (PLAY_SPACE_WIDTH -
PLAY_SPACE_NOBUILD_WIDTH)
|| _position.y < PLAY_SPACE_NOBUILD_HEIGHT || _position.y > (PLAY_SPACE_HEIGHT -
PLAY_SPACE_NOBUILD_HEIGHT))
return GRID_OCCUPIED;

_position.setTo(vo.baseX * GRID_SIZE, vo.baseY * GRID_SIZE);
var hashTL:int = getSmallHash(_position.x, _position.y);
var hashBR:int = getSmallHash(_position.x + ((vo.sizeX - 1) * GRID_SIZE), _position.y +
((vo.sizeY - 1) * GRID_SIZE));
var len:int = hashBR % COLUMNS_SMALL;
var i:int = hashTL;
var fill:int = (remove) ? target : (vo.itemClass == TypeEnum.PYLON) ? 3 : target + 1;
while (i <= hashBR)
{
if (remove)
{
if (_space[i])
{
_space[i] = fill;
}
} else if (_space[i] == target || (_space[i] == null && target == OPEN_SPACE))
{
if (!checking)
{
_space[i] = fill;
}
} else if (!checking && ignoreChecks && (_space[i] == null || _space[i] < fill))
_space[i] = fill;
else if (!ignoreChecks)
return GRID_OCCUPIED;
if (i % COLUMNS_SMALL == len)
{
hashTL += COLUMNS_SMALL;
i = hashTL;
} else
i++;
}
```

```
return ALL_CLEAR;
}

private function placeBase():void
{
//add the no build area
buildGrid(197690, 242450, BUILDING); //201420, 238720, 2);
//243196;325

//add the buildable space for buildings
buildGrid(205150, 238720, PLATFORM);

//add back the open space around the starbase since it is an odd shape
buildGrid(208865, 234204, OPEN_SPACE);
buildGrid(197705, 204444, OPEN_SPACE);
buildGrid(208920, 234259, OPEN_SPACE);
buildGrid(238680, 245419, OPEN_SPACE);
}

private function buildGrid( hashTL:int, hashBR:int, value:int ):void
{
var len:int = hashBR % COLUMNS_SMALL;
var i:int = hashTL;
while (i <= hashBR)
{
_space[i] = value;
if (i % COLUMNS_SMALL == len)
{
hashTL += COLUMNS_SMALL;
i = hashTL;
} else
i++;
}
}


//==========================================================================
//**************************************************************************************************
// ISO Conversions
//**************************************************************************************************

//==========================================================================

//determine the size of the square that we would need to fill the visible view area in iso
public function initIso( width:int, height:int ):void
{
PIVOT.setTo(width / 2, height / 2);
var p:Point = new Point(0, 0);
convertPointFromIsometric(p);
```

```
REAL_OFFSET.setTo(Math.abs(p.x), Math.abs(p.x));
p.setTo(width, height);
convertPointFromIsometric(p);
REAL_SIZE = p.x + REAL_OFFSET.x;
}

//Converts the passed grid-based point to an isometric point.
public function convertPointToIsometric( point:Point ):void
{
var p:Point = point.subtract(REAL_OFFSET)

p = p.subtract(PIVOT);

//Do the actual rotation with the point.
rotate(p, Math.cos(ISO_ROTATION), Math.sin(ISO_ROTATION));

//Scale the Y position so it looks like it's squished into the distance.
p.y *= ISO_SCALE;

//Now we want to go back to the top left corner (we don't want to start from the pivot).
p = p.add(PIVOT);

point.setTo(p.x, p.y);
}

//Converts the passed isometric point to a grid-based point.
//This is literally exactly the same except everything is done in reverse order and opposite
operations.
public function convertPointFromIsometric( point:Point ):void
{
//Go from the corner to the pivot.
var p:Point = point.subtract(PIVOT);

//Unscale the Y position so it's back to grid proportions.
p.y /= ISO_SCALE;

//Do the actual rotation with the point.
rotate(p, Math.cos(ISO_ROTATION), -Math.sin(ISO_ROTATION));

//Now that we're back to normal, un-apply the pivot.
p = p.add(PIVOT);

p = p.add(REAL_OFFSET);

point.setTo(p.x, p.y);
}

public function convertIsoToGrid( point:Point ):void
{
convertPointFromIsometric(point);
```

```
point.x = (point.x / GRID_SIZE) | 0;
point.y = (point.y / GRID_SIZE) | 0;
}

public function convertGridToIso( point:Point ):void
{
point.x = point.x * GRID_SIZE;
point.y = point.y * GRID_SIZE;
convertPointToIsometric(point);
}

public function convertBuildingGridToIso( point:Point, vo:BuildingVO ):void
{
if (vo.prototype)
{
if (vo.itemClass == TypeEnum.FORCEFIELD)
{
point.x = (vo.baseX + 5 * .5);
point.y = (vo.baseY + 5 * .5);
} else
{
point.x = (vo.baseX + vo.sizeX * .5);
point.y = (vo.baseY + vo.sizeY * .5);
}
convertGridToIso(point);
}
}

public function convertBuildingIsoToGrid( point:Point, vo:BuildingVO ):void
{
if (vo.prototype)
{
var p:Point = new Point(point.x, point.y);
convertPointFromIsometric(p);
//building is centered... get the top left corner of the building base
p.x -= vo.sizeX * .49 * GRID_SIZE;
p.y -= vo.sizeY * .49 * GRID_SIZE;
convertPointToIsometric(p);
convertIsoToGrid(p);
vo.baseX = p.x;
vo.baseY = p.y;
}
}

private function rotate( point:Point, cosine:Number, sine:Number ):void
{
var newX:Number = point.x * cosine - point.y * sine;
var newY:Number = point.x * sine + point.y * cosine;

point.x
```

```
= newX;
point.y = newY;
}

public function destroy():void
{
_space = null;
_columnSpan = null;
_rowSpan = null;
_testSpan = null;
}
}
}

-----------------------------------
File 349: igw\com\model\starbase\StarbaseModel.as
package com.model.starbase
{
import com.enum.CurrencyEnum;
import com.model.Model;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.BaseData;
import com.service.server.incoming.data.BuffData;
import com.service.server.incoming.data.BuildingData;
import com.service.server.incoming.data.ResearchData;
import com.service.server.incoming.data.TradeRouteData;

import flash.utils.Dictionary;

import org.shared.ObjectPool;

/**
* Keeps tracks of the player's bases.
*/
public class StarbaseModel extends Model
{
public var entryData:*;
public var entryView:Class; //the view to show upon entering the starbase

private var _bases:Dictionary;
private var _currentBase:BaseVO;
private var _tempBase:BaseVO;

[PostConstruct]
public function init():void
{
_bases = new Dictionary();
}


//=============================================================================
```

```
//**************************************************************************************************************
// BASES
//**************************************************************************************************************

//==========================================================================

public function importBaseData( baseData:BaseData, initTradeRoutes:Boolean = true ):void
{
if (!_bases[baseData.id])
{
var baseVO:BaseVO = ObjectPool.get(BaseVO);
baseVO.init(baseData);
if (initTradeRoutes)
baseVO.initTradeRoutes();
_bases[baseData.id] = baseVO;
}
if (_currentBase == null)
_currentBase = _bases[baseData.id];
_bases[baseData.id].importData(baseData);
}

public function getBaseByID( id:String ):BaseVO { return _bases[id]; }

public function switchBase( baseID:String ):void
{
_tempBase = _bases[baseID];
if (_tempBase)
_currentBase = _tempBase;
}

public function setBaseDirty():void
{
for each (var base:BaseVO in _bases)
{
base.dirty = true;
}
}

public function updateBases():void
{
for each (var base:BaseVO in _bases)
{
base.dirty = true;
}
}

public function isBaseDamaged( baseID:String = null ):Boolean
{
var baseVO:BaseVO = (baseID != null) ? getBaseByID(baseID) : _currentBase;
for
```

```
(var i:int = 0; i < baseVO.buildings.length; i++)
{
if (baseVO.buildings[i].currentHealth != 1)
return true;
}
return false;
}

public function removeBaseByID( id:String ):void
{
if (_bases[id])
{
ObjectPool.give(_bases[id]);
delete _bases[id];
_bases[id] = null;
}
}


//=============================================================================
//*****************************************************************************
// BUILDINGS
//*****************************************************************************

//=============================================================================

public function importBuildingData( buildingData:BuildingData ):void
{
_tempBase = _bases[buildingData.baseID];
if (_tempBase)
_tempBase.importBuildingData(buildingData);
}

public function addBuilding( buildingVO:BuildingVO ):void
{
_tempBase = _bases[buildingVO.baseID];
if (_tempBase)
_tempBase.addBuilding(buildingVO);
}

public function updateBuildingID( oldID:String, newID:String ):void
{
//ensure the building exists
var buildingVO:BuildingVO = getBuildingByID(oldID, false);
if (buildingVO)
{
//update its' id
_tempBase = _bases[buildingVO.baseID];
_tempBase.updateBuildingID(oldID, newID);
}
}
```

```
public function getBuildingByID( id:String, useCurrentBase:Boolean = true ):BuildingVO
{
var buildingVO:BuildingVO;
if (useCurrentBase)
buildingVO = _currentBase.getBuildingByID(id);
else
{
for each (var base:BaseVO in _bases)
{
buildingVO = base.getBuildingByID(id);
if (buildingVO)
break;
}
}
return buildingVO;
}

public function getBuildingsByBaseID( id:String = null ):Vector.<BuildingVO>
{
_tempBase = (id == null) ? _currentBase : _bases[id];
if (_tempBase)
return _tempBase.buildings;
return null;
}

public function getBuildingByClass( buildingClass:String, highestLevel:Boolean = false
):BuildingVO
{
var buildingVO:BuildingVO = _currentBase.getBuildingByClass(buildingClass, highestLevel);
if (buildingVO)
return buildingVO;
return null;
}

public function removeBuildingByID( id:String ):BuildingVO
{
var buildingVO:BuildingVO = getBuildingByID(id, false);
if (buildingVO)
_bases[buildingVO.baseID].removeBuilding(buildingVO);
return buildingVO;
}


//==========================================================================
//**************************************************************************************************
// RESEARCH
//**************************************************************************************************

//==========================================================================
```

```
public function importResearchData( researchData:ResearchData ):void
{
_tempBase = _bases[researchData.baseID];
if (_tempBase)
_tempBase.importResearchData(researchData);
}

public function addBeginnerResearch( research:Vector.<IPrototype> ):void
{
var researchData:ResearchData = ObjectPool.get(ResearchData);
for (var i:int = 0; i < research.length; i++)
{
if (research[i].getValue('requiredBuilding') == '')
{
for each (var base:BaseVO in _bases)
{
researchData.baseID = base.id;
researchData.id = 'Research' + i;
researchData.playerOwnerID = CurrentUser.id;
researchData.prototype = research[i];
importResearchData(researchData);
}
}
}
ObjectPool.give(researchData);
}

public function getAllStarbaseResearchTransactions():Vector.<ResearchVO>
{
var v:Vector.<ResearchVO> = new Vector.<ResearchVO>();

for each (var base:BaseVO in _bases)
v.concat(base.research);

return v;
}

public function getResearchByID( id:String, useCurrentBase:Boolean = true ):ResearchVO
{
var researchVO:ResearchVO;
if (useCurrentBase)
researchVO = _currentBase.getResearchByID(id);
else
{
for each (var base:BaseVO in _bases)
{
researchVO = base.getResearchByID(id);
if (researchVO)
break;
```

```
}
}
return researchVO;
}

public function removeResearchByID( id:String ):ResearchVO
{
var researchVO:ResearchVO = getResearchByID(id, false);
if (researchVO)
_bases[researchVO.baseID].removeResearch(researchVO);
return researchVO;
}

public function isResearched( name:String ):Boolean
{
var baseVO:BaseVO = currentBase;
if (name == '' || name == null || baseVO.reqsDisabled != 0)
return true;
var researched:Vector.<ResearchVO> = research;
for (var i:int = 0; i < researched.length; i++)
{
if (researched[i].name == name)
return true;
}
return false;
}

public function get research():Vector.<ResearchVO> { return _currentBase ?
_currentBase.research : null; }


//=========================================================================
//*************************************************************************************************
// TRADE ROUTES
//*************************************************************************************************

//=========================================================================

public function importTradeRouteData( tradeRouteData:TradeRouteData ):void
{
_tempBase = _bases[tradeRouteData.baseID];
if (_tempBase)
_tempBase.importTradeRouteData(tradeRouteData);
}

public function updateTradeRouteID( oldID:String, newID:String ):void
{
//ensure the traderoute exists
var tradeRouteVO:TradeRouteVO = getTradeRouteByID(oldID, false);
if
```

```
(tradeRouteVO)
{
_tempBase = _bases[tradeRouteVO.baseID];
_tempBase.updateTradeRouteID(oldID, newID);
}
}

public function getTradeRouteByID( id:String, useCurrentBase:Boolean = true ):TradeRouteVO
{
var tradeRouteVO:TradeRouteVO;
if (useCurrentBase)
tradeRouteVO = _currentBase.getTradeRouteByID(id);
else
{
for each (var base:BaseVO in _bases)
{
tradeRouteVO = base.getTradeRouteByID(id);
if (tradeRouteVO)
break;
}
}
return tradeRouteVO;
}

public function getTradeRouteByCorporation( corporation:String, useCurrentBase:Boolean =
true ):TradeRouteVO
{
var tradeRouteVO:TradeRouteVO;
if (useCurrentBase)
tradeRouteVO = _currentBase.getTradeRouteByCorporation(corporation);
else
{
for each (var base:BaseVO in _bases)
{
tradeRouteVO = base.getTradeRouteByCorporation(corporation);
if (tradeRouteVO)
break;
}
}
return tradeRouteVO;
}

public function getTradeRoutesByBaseID( id:String = null ):Vector.<TradeRouteVO>
{
_tempBase = (id == null) ? _currentBase : _bases[id];
if (_tempBase)
return _tempBase.tradeRoutes;
return null;
}

public
```

```
function removeTradeRouteByID( id:String ):TradeRouteVO
{
var tradeRouteVO:TradeRouteVO = getTradeRouteByID(id, false);
if (tradeRouteVO)
_bases[tradeRouteVO.baseID].removeTradeRoute(tradeRouteVO);
return tradeRouteVO;
}


//==========================================================================================
//******************************************************************************************
// BUFFS
//******************************************************************************************

//==========================================================================================

public function importBuffData( buffData:BuffData ):void
{
_tempBase = _bases[buffData.baseID];
if (_tempBase)
_tempBase.importBuffData(buffData);
}

public function getBuffByID( id:String, useCurrentBase:Boolean = true ):BuffVO
{
var buffVO:BuffVO;
if (useCurrentBase)
buffVO = _currentBase.getBuffByID(id);
else
{
for each (var base:BaseVO in _bases)
{
buffVO = base.getBuffByID(id);
if (buffVO)
break;
}
}
return buffVO;
}

public function getBuffByType( type:String, useCurrentBase:Boolean = true ):BuffVO
{
var buffVO:BuffVO;
if (useCurrentBase)
buffVO = _currentBase.getBuffByType(type);
else
{
for each (var base:BaseVO in _bases)
{
buffVO = base.getBuffByType(type);
if
```

```
(buffVO)
break;
}
}
return buffVO;
}

public function getBuffsByBaseID( id:String = null ):Vector.<BuffVO>
{
_tempBase = (id == null) ? _currentBase : _bases[id];
if (_tempBase)
return _tempBase.buffs;
return null;
}

public function updateBuffID( oldID:String, newID:String ):void
{
//ensure the building exists
var buffVO:BuffVO = getBuffByID(oldID, false);
if (buffVO)
{
//update its' id
_tempBase = _bases[buffVO.baseID];
_tempBase.updateBuffID(oldID, newID);
}
}

public function removeBuffByID( id:String ):BuffVO
{
var buffVO:BuffVO = getBuffByID(id, false);
if (buffVO)
_bases[buffVO.baseID].removeBuff(buffVO);
return buffVO;
}

public function getCurrentResourceCount( type:String ):uint
{
if (_currentBase)
{
var resource:uint = 0;
switch (type)
{
case CurrencyEnum.ALLOY:
resource = _currentBase.alloy;
break;
case CurrencyEnum.CREDIT:
resource = _currentBase.credits;
break;
case CurrencyEnum.ENERGY:
resource = _currentBase.energy;
break;
```

```
case CurrencyEnum.SYNTHETIC:
resource = _currentBase.synthetic;
break;
}
return resource;
} else
return 0;
}

public function addListener( callback:Function ):void {
_currentBase.onResourcesChange.add(callback); }
public function removeListener( callback:Function ):void {
_currentBase.onResourcesChange.remove(callback); }

public function get baseCreditIncome():uint { return _currentBase ?
_currentBase.baseCreditIncome : null; }
public function get baseResourceIncome():uint { return _currentBase ?
_currentBase.baseResourceIncome : null; }

public function get baseResourcePurchaseScale():Number { return _currentBase ?
_currentBase.baseResourcePurchaseScale : 0; }
public function get baseCreditPurchaseScale():Number { return _currentBase ?
_currentBase.baseCreditPurchaseScale : 0; }


public function get tradeRouteCreditIncome():uint { return _currentBase ?
_currentBase.tradeRouteCreditIncome : null; }
public function get tradeRouteResourceIncome():uint { return _currentBase ?
_currentBase.tradeRouteResourceIncome : null; }

public function get buildings():Vector.<BuildingVO> { return _currentBase ?
_currentBase.buildings : null; }

public function get currentBase():BaseVO { return _currentBase; }
public function get currentBaseID():String { return _currentBase.id; }

public function get homeBase():BaseVO { return _bases[CurrentUser.homeBase]; }
public function get centerSpaceBase():BaseVO
{
if (CurrentUser.centerSpaceBase)
return _bases[CurrentUser.centerSpaceBase];
return null;
}

public function get maxCredits():uint { return _currentBase ? _currentBase.maxCredits : 0; }
public function get maxResources():uint { return _currentBase ? _currentBase.maxResources :
0; }

public function get grid():StarbaseGrid { return _currentBase.grid; }
}
```

```
}



-----------------------------------
File 350: igw\com\model\starbase\TradeRouteVO.as
package com.model.starbase
{
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.TradeRouteData;

import flash.events.TimerEvent;
import flash.utils.Timer;

public class TradeRouteVO
{
private static var _tempID:int = 0;

//NEW
private var _baseID:String;
private var _contractGroup:String;
private var _factionPrototype:IPrototype;
private var _id:String;
private var _name:String;
private var _reputation:Number;

// current contract params
private var _pointValueRolled:Number;
private var _offerExpires:uint;
private var _contractPrototype:IPrototype;
private var _productivity:Number;
private var _payout:Number;
private var _duration:Number;
private var _frequency:Number;
private var _security:Number;
private var _bribe:Number;
private var _offerRejected:Boolean;
private var _reputationWhenBegun:Number;
private var _began:uint;
private var _end:uint;

//Resource Modifiers
private var _creditScale:Number;
private var _alloyScale:Number;
private var _energyScale:Number;
private var _syntheticScale:Number;
private var _reputationScale:Number;

//Resource
```

```
Per Second
private var _creditsPerSecond:Number;
private var _alloyPerSecond:Number;
private var _energyPerSecond:Number;
private var _syntheticPerSecond:Number;
private var _reputationPerSecond:Number;

public function init( id:String, factionPrototype:IPrototype = null ):void
{
_id = id;
_factionPrototype = factionPrototype;
reset();
}

public function importData( tradeRouteData:TradeRouteData ):void
{
_baseID = tradeRouteData.baseID;
_began = tradeRouteData.began;
_bribe = tradeRouteData.bribe;
_factionPrototype = tradeRouteData.factionPrototype;
_contractGroup = _factionPrototype.getValue('contractGroup');
_contractPrototype = tradeRouteData.contractPrototype;
_duration = tradeRouteData.duration;
_end = tradeRouteData.end;
_frequency = tradeRouteData.frequency;
_name = tradeRouteData.name;
_offerExpires = tradeRouteData.offerExpires;
_offerRejected = tradeRouteData.offerRejected
_payout = tradeRouteData.payout;
_productivity = tradeRouteData.productivity;
_reputation = tradeRouteData.reputation;
_reputationWhenBegun = tradeRouteData.reputationWhenBegun;
_security = tradeRouteData.security;

if (_contractPrototype != null)
{
_creditScale = _contractPrototype.getValue('creditScale');
_alloyScale = _contractPrototype.getValue('alloyScale');
_energyScale = _contractPrototype.getValue('energyScale');
_syntheticScale = _contractPrototype.getValue('syntheticsScale');
_reputationScale = _contractPrototype.getValue('reputationScale');
}
}

public function reset():void
{
if (_factionPrototype)
_contractGroup = _factionPrototype.getValue('contractGroup');
_pointValueRolled = 0;
_offerExpires = 0;
_contractPrototype
```

```
= null;
_productivity = 0;
_payout = 0;
_duration = 0;
_frequency = 0;
_security = 0;
_began = 0;
_end = 0;

_creditScale = 0;
_alloyScale = 0;
_energyScale = 0;
_syntheticScale = 0;
_reputationScale = 0;

_bribe = 0;

_offerRejected = true;
_reputationWhenBegun = 0;

_creditsPerSecond = 0;
_alloyPerSecond = 0;
_energyPerSecond = 0;
_syntheticPerSecond = 0;
_reputationPerSecond = 0;
}

public function clone():TradeRouteVO
{
var vo:TradeRouteVO = new TradeRouteVO();
var data:TradeRouteData = new TradeRouteData();
data.factionPrototype = _factionPrototype;
data.id = TEMP_ID;
data.baseID = _baseID;
data.reputation = _reputation;
data.contractPrototype = _contractPrototype;
data.offerExpires = _offerExpires;
data.payout = _payout;
data.duration = _duration;
data.frequency = _frequency;
data.security = _security;
data.began = _began;
data.end = _end;
vo.importData(data);
vo.pointValueRolled = _pointValueRolled;

return vo;
}

internal function forceSetID( v:String ):void { _id = v; }

public
```

```
function get baseID():String { return _baseID; }
public function set baseID( baseID:String ):void { _baseID = baseID; }
public function get bribe():Number { return _bribe; }
public function get contractGroup():String { return _contractGroup; }
public function get name():String { return _name; }
public function get offerExpires():Number { return _offerExpires; }
public function set offerExpires( offerExpires:Number ):void { _offerExpires = offerExpires; }
public function get pointValueRolled():Number { return _pointValueRolled; }
public function set pointValueRolled( pointValueRolled:Number ):void { _pointValueRolled =
pointValueRolled; }
public function get rejected():Boolean { return _offerRejected; }
public function get reputation():Number { return _reputation; }
public function set reputation( reputation:Number ):void { _reputation = reputation; }
public function get reputationWhenBegun():Number { return _reputationWhenBegun; }

public function set contractPrototype( contractPrototype:IPrototype ):void
{
_contractPrototype = contractPrototype;
if (_contractPrototype != null)
{
_creditScale = _contractPrototype.getValue('creditScale');
_alloyScale = _contractPrototype.getValue('alloyScale');
_energyScale = _contractPrototype.getValue('energyScale');
_syntheticScale = _contractPrototype.getValue('syntheticsScale');
_reputationScale = _contractPrototype.getValue('reputationScale');
}
}

public function get contractPrototype():IPrototype { return _contractPrototype; }
public function get corporation():String { return _factionPrototype.name; }

public function get factionPrototype():IPrototype { return _factionPrototype; }

public function get productivity():Number { return _productivity; }

public function set productivity( productivity:Number ):void { _productivity = productivity; }

public function get payout():Number { return _payout; }

public function set payout( payout:Number ):void { _payout = payout; }

public function get duration():Number { return _duration; }

public function set duration( duration:Number ):void { _duration = duration; }

public function get frequency():Number { return _frequency; }

public function set frequency( frequency:Number ):void { _frequency = frequency; }

public function get security():Number { return _security; }

public
```

```
function set security( security:Number ):void { _security = security; }

public function get began():Number { return _began; }

public function set began( began:Number ):void { _began = began; }

public function get end():Number { return _end; }

public function set end( end:Number ):void { _end = end; }

public function get creditScale():Number { return _creditScale; }

public function get alloyScale():Number { return _alloyScale; }

public function get energyScale():Number { return _energyScale; }

public function get syntheticScale():Number { return _syntheticScale; }

public function get reputationScale():Number { return _reputationScale; }

public function get id():String { return _id; }

public static function get TEMP_ID():String { _tempID++; return "player." + CurrentUser.name +
"." + _tempID }

public function destroy():void
{

}
}
}

-----------------------------------
File 351: igw\com\model\transaction\TransactionModel.as
package com.model.transaction
{
import com.enum.server.RequestEnum;
import com.event.TransactionEvent;
import com.event.signal.TransactionSignal;
import com.model.Model;
import com.service.server.ITransactionRequest;
import com.service.server.ITransactionResponse;

import flash.utils.Dictionary;

import org.shared.ObjectPool;

public class TransactionModel extends Model
{
private var _clientData:Dictionary;
private
```

```
var _id:int;
private var _transactions:Dictionary;
private var _transactionSignal:TransactionSignal;

[PostConstruct]
public function init():void
{
_clientData = new Dictionary();
_id = 0;
_transactions = new Dictionary();
_transactionSignal = new TransactionSignal();
}

public function addTransaction( request:ITransactionRequest, id:String, type:String, data:Object,
createDefaultTransaction:Boolean ):void
{
request.token = token;
data.token = request.token;
data.type = type;

_clientData[request.token] = data;
if (createDefaultTransaction)
{
var transaction:TransactionVO = ObjectPool.get(TransactionVO);
transaction.init(id, type, data.token);
_transactions[data.token] = transaction;
updatedTransaction(transaction);
}
}

public function handleResponse( response:ITransactionResponse ):Object
{
var transactionToken:int = response.token;
response.data.type = getType(response.data.messageID);
//update the _id if the transactionToken is greater so that we don't reuse a id that is already in
use
if (_id <= transactionToken)
_id = transactionToken + 1;
if (_transactions[transactionToken] != null)
{
//update the transaction we have instead of replacing
//object pool the old one and swap in the response
_transactions[transactionToken].importData(response.data);
ObjectPool.give(response.data);
response.data = _transactions[transactionToken];
} else
_transactions[transactionToken] = response.data;
return _clientData[transactionToken];
}

public
```

```
function updatedTransaction( transactionVO:TransactionVO ):void
{
_transactionSignal.dispatch(TransactionSignal.TRANSACTION_UPDATED, transactionVO);
}

public function removeTransaction( tranactionToken:int ):void
{
var transaction:TransactionVO = _transactions[tranactionToken];
_transactions[tranactionToken] = null;
delete _transactions[tranactionToken];
//clear the clientData
_clientData[tranactionToken] = null;
delete _clientData[tranactionToken];
_transactionSignal.dispatch(TransactionSignal.TRANSACTION_REMOVED, transaction);
//object pool the transaction
ObjectPool.give(transaction);
}

public function dataImported():void
{
_transactionSignal.dispatch(TransactionSignal.DATA_IMPORTED, null);
}

public function getTransactionByID( id:String ):TransactionVO
{
for each (var transaction:TransactionVO in _transactions)
{
if (transaction.id == id)
return transaction;
}
return null;
}

public function getTransactionByToken( token:int ):TransactionVO
{
if (_transactions.hasOwnProperty(token))
return _transactions[token];
return null;
}

private function getType( messageID:int ):String
{
var type:String;
switch (messageID)
{
case RequestEnum.STARBASE_REPAIR_FLEET:
type = TransactionEvent.STARBASE_REPAIR_FLEET;
break;
case RequestEnum.STARBASE_BUILD_NEW_BUILDING:
type = TransactionEvent.STARBASE_BUILDING_BUILD;
break;
```

```
case RequestEnum.STARBASE_UPGRADE_BUILDING:
type = TransactionEvent.STARBASE_BUILDING_UPGRADE;
break;
case RequestEnum.STARBASE_RECYCLE_BUILDING:
type = TransactionEvent.STARBASE_BUILDING_RECYCLE;
break;
case RequestEnum.STARBASE_REFIT_BUILDING:
type = TransactionEvent.STARBASE_REFIT_BUILDING;
break;
case RequestEnum.STARBASE_REPAIR_BASE:
type = TransactionEvent.STARBASE_REPAIR_BASE;
break;
case RequestEnum.STARBASE_MOVE_BUILDING:
type = TransactionEvent.STARBASE_BUILDING_MOVE;
break;
case RequestEnum.STARBASE_RESEARCH:
type = TransactionEvent.STARBASE_RESEARCH;
break;
case RequestEnum.STARBASE_BUY_STORE_ITEM:
type = TransactionEvent.STARBASE_BUY_STORE_ITEM;
break;
case RequestEnum.STARBASE_BUY_OTHER_STORE_ITEM:
type = TransactionEvent.STARBASE_BUY_OTHER_STORE_ITEM;
break;
case RequestEnum.STARBASE_BUY_RESOURCE:
type = TransactionEvent.STARBASE_BUY_RESOURCES;
break;
case RequestEnum.STARBASE_BUILD_SHIP:
type = TransactionEvent.STARBASE_BUILD_SHIP;
break;
case RequestEnum.STARBASE_RECYCLE_SHIP:
type = TransactionEvent.STARBASE_RECYCLE_SHIP;
break;
case RequestEnum.STARBASE_RECALL_FLEET:
type = TransactionEvent.STARBASE_RECALL_FLEET;
break;
case RequestEnum.STARBASE_REFIT_SHIP:
type = TransactionEvent.STARBASE_REFIT_SHIP;
break;
case RequestEnum.STARBASE_NEGOTIATE_CONTRACT:
type = TransactionEvent.STARBASE_NEGOTIATE_CONTRACT_REQUEST;
break;
case RequestEnum.STARBASE_UPDATE_FLEET:
type = TransactionEvent.STARBASE_UPDATE_FLEET;
/* TODO -- ?
public static const STARBASE_BRIBE_CONTRACT:int = 23;
public static const STARBASE_CANCEL_CONTRACT:int = 24;
public static const STARBASE_EXTEND_CONTRACT:int = 25;
public static const STARBASE_RESECURE_CONTRACT:int = 26;
*/
```

```
}

return type;
}

public function addListener( type:int, listener:Function ):void { _transactionSignal.add(type,
listener); }
public function removeListener( listener:Function ):void { _transactionSignal.remove(listener); }

public function get token():int { return _id++; }
public function get transactions():Dictionary { return _transactions; }
}
}
```

File 352: igw\com\model\transaction\TransactionVO.as

```
package com.model.transaction
{
import com.enum.server.StarbaseTransactionStateEnum;
import com.service.server.BinaryInputStream;

import flash.utils.getTimer;

public class TransactionVO
{
public var id:String;
public var type:String;

private var _baseID:String;
private var _began:uint;
private var _clientTime:int;
private var _ends:uint;
private var _messageID:int;
private var _reason:String;
private var _serverKey:String;
private var _state:int;
private var _success:Boolean;
private var _temp:Number;
private var _timeMS:Number;
private var _timeRemainingMS:Number;
private var _token:int;

public function init( id:String, type:String, token:int ):void
{
this.id = id;
this.type = type;
_token = token;
setPendingState();
```

```
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();
_token = input.readInt();
_serverKey = input.readUTF();
id = input.readUTF();
_messageID = input.readByte(); // should match the original message used to create this
transaction
_success = input.readBoolean();
_state = input.readInt(); // see StarbaseTransactionStateEnum
_timeMS = input.readInt(); // Total time Cost
_began = input.readInt64(); // Begin Time Stamp
_ends = input.readInt64(); //End Time Stamp
_timeRemainingMS = input.readInt(); // Total time left
_baseID = input.readUTF();
_reason = input.readUTF(); // debug string, will probably go away eventually
input.checkToken();

_clientTime = getTimer();
}

public function readJSON( data:Object ):void
{
_baseID = data.basePersistence;
_began = data.began; // Begin Time Stamp
_clientTime = getTimer();
_ends = data.ends; //End Time Stamp
id = data.key;
_messageID = data.messageId; // should match the original message used to create this
transaction
_reason = data.reason; // debug string, will probably go away eventually
_serverKey = data.transactionKey;
_state = data.state; // see StarbaseTransactionStateEnum
_success = data.succes;
_timeMS = data.time_cost_milliseconds; // Total time Cost
_timeRemainingMS = data.timeRemainingMS; // Total time left
_token = data.token;
}

public function importData( vo:TransactionVO ):void
{
_baseID = vo.baseID;
_began = vo.began;
_clientTime = getTimer();
_ends = vo.ends;
id = vo.id;
_messageID = vo.messageID;
_reason
```

```
= vo.reason;
_serverKey = vo.serverKey;
_state = vo.state;
_success = vo.success;
_timeMS = vo.timeMS;
_timeRemainingMS = vo.serverTimeRemainingMS;
_token = vo.token;
}

public function setPendingState():void
{
_began = _ends = _clientTime = 0;
_timeMS = _timeRemainingMS = -1;
_state = StarbaseTransactionStateEnum.PENDING;
}

public function get baseID():String { return _baseID; }

public function get began():uint { return _began; }

public function get ends():uint { return _ends; }

public function get messageID():int { return _messageID; }

public function get percentComplete():Number
{
var percent:Number = 1 - (timeRemainingMS / _timeMS);
if (percent < 0)
percent = 0;
return (percent > 1) ? 1 : percent;
}

public function get reason():String { return _reason; }

public function get serverKey():String { return _serverKey; }

public function get state():int { return _state; }

public function get success():Boolean { return _success; }

public function get timeMS():Number { return _timeMS; }

public function get serverTimeRemainingMS():Number { return _timeRemainingMS; }

public function get timeRemainingMS():Number
{
if (_state == StarbaseTransactionStateEnum.PENDING)
return -1;
_temp = _timeRemainingMS - (getTimer() - _clientTime);
if (_temp < 0)
_temp
```

```
= 0;
return _temp;
}

public function get token():int { return _token; }

public function destroy():void
{

}
}
}

-----------------------------------
File 353: igw\com\model\warfrontModel\WarfrontModel.as
package com.model.warfrontModel
{
import com.model.Model;
import com.model.prototype.PrototypeModel;
import com.service.server.incoming.leaderboard.WarfrontUpdateResponse;

import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class WarfrontModel extends Model
{
private var _battles:Vector.<WarfrontVO>;
private var _i:int;
private var _lookup:Dictionary;
private var _prototypeModel:PrototypeModel
private var _updateSignal:Signal;

public function WarfrontModel()
{
_battles = new Vector.<WarfrontVO>;
_lookup = new Dictionary();
_updateSignal = new Signal(Vector.<WarfrontVO>, Vector.<String>);
}

public function importData( response:WarfrontUpdateResponse ):void
{
var battle:WarfrontVO;
//add new warfronts
for (_i = 0; _i < response.warfronts.length; _i++)
{
battle = ObjectPool.get(WarfrontVO);
battle.importData(response.warfronts[_i]);
_battles.unshift(battle);
_lookup[battle.id]
```

```
= battle;
}

//remove warfronts
var index:int;
for (_i = 0; _i < response.removed.length; _i++)
{
battle = _lookup[response.removed[_i]];
if (battle)
{
index = _battles.indexOf(battle);
if (index > -1)
{
_battles.splice(index, 1);
ObjectPool.give(battle);
}
delete _lookup[battle.id];
}
}

_updateSignal.dispatch(_battles, response.removed);
}

public function addUpdateListener( listener:Function ):void { _updateSignal.add(listener); }
public function removeUpdateListener( listener:Function ):void { _updateSignal.remove(listener);
}

public function get battles():Vector.<WarfrontVO> { return _battles; }

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
}
}

-----------------------------------
File 354: igw\com\model\warfrontModel\WarfrontVO.as
package com.model.warfrontModel
{
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.WarfrontData;

public class WarfrontVO
{
public var id:String;
public var battleServerAddress:String;

public var attackerID:String;
public var attackerName:String;
public var attackerRace:IPrototype;
public var attackerFleetRating:int;

public
```

```
var defenderID:String;
public var defenderName:String;
public var defenderRace:IPrototype;
public var defenderFleetRating:int;
public var defenderBaseRating:int;

public var sector:String;
public var sectorX:Number;
public var sectorY:Number;

public function importData( data:WarfrontData ):void
{
id = data.id;
battleServerAddress = data.battleServerAddress;

attackerID = data.attackerID;
attackerName = data.attackerName;
attackerRace = data.attackerRace;
attackerFleetRating = data.attackerFleetRating;

defenderID = data.defenderID;
defenderName = data.defenderName;
defenderRace = data.defenderRace;
defenderFleetRating = data.defenderFleetRating;
defenderBaseRating = data.defenderBaseRating;

sector = data.sector;
sectorX = data.sectorX;
sectorY = data.sectorY;
}

public function destroy():void
{
attackerRace = null;
defenderRace = null;
}
}
}

-------------------------------------
File 355: igw\com\presenter\IImperiumPresenter.as
package com.presenter
{
import org.robotlegs.extensions.presenter.api.IPresenter;

public interface IImperiumPresenter extends IPresenter
{
function playSound( sound:String, volume:Number = 0.5 ):void;

function addStateListener( callback:Function ):Boolean;
function
```

```
removeStateListener( callback:Function ):Boolean;

function get hudEnabled():Boolean;
function set hudEnabled( value:Boolean ):void;

function get inFTE():Boolean;
}
}

-----------------------------------
File 356: igw\com\presenter\ImperiumPresenter.as
package com.presenter
{
import com.controller.fte.FTEController;
import com.controller.sound.SoundController;
import com.event.StateEvent;

import org.osflash.signals.Signal;
import org.robotlegs.extensions.presenter.impl.Presenter;

public class ImperiumPresenter extends Presenter implements IImperiumPresenter
{
protected static var _hudEnabled:Boolean = true;

protected var _fteController:FTEController;
protected var _soundController:SoundController;
protected var _stateChangeSignal:Signal;

[PostConstruct]
override public function init():void
{
super.init();
_stateChangeSignal = new Signal(String);

addContextListener(StateEvent.GAME_STARBASE_CLEANUP, onStateChange, StateEvent);
addContextListener(StateEvent.GAME_BATTLE_CLEANUP, onStateChange, StateEvent);
addContextListener(StateEvent.GAME_SECTOR_CLEANUP, onStateChange, StateEvent);
addContextListener(StateEvent.GAME_STARBASE, onStateChange, StateEvent);
addContextListener(StateEvent.GAME_BATTLE, onStateChange, StateEvent);
addContextListener(StateEvent.GAME_SECTOR, onStateChange, StateEvent);
}

public function playSound( sound:String, volume:Number = 0.5 ):void {
_soundController.playSound(sound, volume); }

public function addStateListener( callback:Function ):Boolean {
_stateChangeSignal.add(callback); return true; }
public function removeStateListener( callback:Function ):Boolean {
_stateChangeSignal.remove(callback); return true; }

protected
```

```
function removeContextListeners():Boolean
{
removeContextListener(StateEvent.GAME_STARBASE_CLEANUP, onStateChange,
StateEvent);
removeContextListener(StateEvent.GAME_BATTLE_CLEANUP, onStateChange, StateEvent);
removeContextListener(StateEvent.GAME_SECTOR_CLEANUP, onStateChange, StateEvent);
removeContextListener(StateEvent.GAME_STARBASE, onStateChange, StateEvent);
removeContextListener(StateEvent.GAME_BATTLE, onStateChange, StateEvent);
removeContextListener(StateEvent.GAME_SECTOR, onStateChange, StateEvent);
return true;
}

protected function onStateChange( e:StateEvent ):void
{
_hudEnabled = true;
_stateChangeSignal.dispatch(e.type);
}

public function get hudEnabled():Boolean { return _hudEnabled; }
public function set hudEnabled( value:Boolean ):void { _hudEnabled = value; }

public function get inFTE():Boolean { return _fteController.running; }

[Inject]
public function set fteController( v:FTEController ):void { _fteController = v; }
[Inject]
public function set soundController( v:SoundController ):void { _soundController = v; }

override public function destroy():void
{
removeContextListeners();
super.destroy();

_fteController = null;
_soundController = null;
_stateChangeSignal.removeAll();
_stateChangeSignal = null;
}
}
}

-------------------------------------
File 357: igw\com\presenter\battle\BattlePresenter.as
package com.presenter.battle
{
import com.controller.ServerController;
import com.controller.transaction.TransactionController;
import com.enum.CategoryEnum;
import com.event.SectorEvent;
import com.event.StarbaseEvent;
import
```

```
com.event.StateEvent;
import com.game.entity.components.battle.Modules;
import com.game.entity.components.shared.Position;
import com.game.entity.systems.battle.VitalsSystem;
import com.game.entity.systems.interact.BattleInteractSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.battle.BattleEntityVO;
import com.model.battle.BattleModel;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintModel;
import com.model.blueprint.BlueprintVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.model.sector.SectorModel;
import com.presenter.shared.GamePresenter;
import com.service.ExternalInterfaceAPI;

import flash.geom.Point;
import flash.utils.Dictionary;

import org.ash.core.Entity;
import org.osflash.signals.Signal;

public class BattlePresenter extends GamePresenter implements IBattlePresenter
{
private var _abilitySignal:Signal;
private var _battleModel:BattleModel;
private var _blueprintModel:BlueprintModel;
private var _transactionController:TransactionController;
private var _fleetModel:FleetModel;
private var _playerModel:PlayerModel;
private var _sectorModel:SectorModel;
private var _serverController:ServerController;
private var _startSignal:Signal;
private var _system:BattleInteractSystem;

[PostConstruct]
override public function init():void
{
super.init();

_abilitySignal = new Signal(Entity);
participants.sort(orderParticipants);
_startSignal = new Signal();
_system = BattleInteractSystem(_game.getSystem(BattleInteractSystem));
_system.presenter
```

```
= this;
}

public function getModuleAssetVo( entity:Entity, idx:Number ):AssetVO
{
var modules:Modules = entity.get(Modules);
var abilityPrototype:IPrototype = modules.activatedModules[idx];
return _assetModel.getEntityData(abilityPrototype.uiAsset);
}

public function loadSmallImage( portraitName:String, callback:Function ):void
{
if (portraitName != '')
{
var avatarVO:AssetVO = AssetVO(AssetModel.instance.getFromCache(portraitName));
if (avatarVO)
AssetModel.instance.getFromCache('assets/' + avatarVO.smallImage, callback);
}
}

public function loadMediumImage( portraitName:String, callback:Function ):void
{
if (portraitName != '')
{
var avatarVO:AssetVO = AssetVO(AssetModel.instance.getFromCache(portraitName));
if (avatarVO)
AssetModel.instance.getFromCache('assets/' + avatarVO.mediumImage, callback);
}
}

public function addStartListener( callback:Function ):void { _startSignal.addOnce(callback); }
public function removeStartListener( callback:Function ):void { _startSignal.remove(callback); }

public function onBattleStarted():void
{
if (_startSignal)
_startSignal.dispatch();
//update the fte if it is running
if (_fteController && _fteController.running)
_fteController.nextStep();
}
public function onBattleEnded():void
{
_startSignal.dispatch();
if (BattleInteractSystem(_system).inFTE && _fteController.running)
{
_system.inFTE = false;
_fteController.nextStep();
}
}

public
```

```
function sharePvPVictory():void
{
var enemyVO:PlayerVO = getPlayer(participants[1]);
ExternalInterfaceAPI.shareVictory(enemyVO, isBaseCombat);
}

public function getSelectedFleet():FleetVO
{
return _fleetModel.getFleet(_sectorModel.focusFleetID);
}

public function getShip( shipId:String ):Entity
{
return _game.getEntity(shipId);
}

public function selectOwnedShipById( shipId:String ):void
{
_system.selectOwnedShipByID(shipId);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function loadMiniIconFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
var icon:String = _currentAssetVO.iconImage;
loadIcon(icon, callback);
}

public function addListenerVitalPercentUpdates( listener:Function ):void
{
var vitalsSystem:VitalsSystem = VitalsSystem(_game.getSystem(VitalsSystem));
if (vitalsSystem)
vitalsSystem.onHealthChanged.add(listener);
}

public function removeListenerVitalPercentUpdates( listener:Function ):void
{
var vitalsSystem:VitalsSystem = VitalsSystem(_game.getSystem(VitalsSystem));
if (vitalsSystem)
vitalsSystem.onHealthChanged.remove(listener);
}

public function getHealthPercentByPlayerID( id:String ):Number
{
var vitalsSystem:VitalsSystem = VitalsSystem(_game.getSystem(VitalsSystem));
if
```

```
(vitalsSystem)
return vitalsSystem.getTotalHealthByPlayer(id);

return 1;
}

public function getPlayer( id:String ):PlayerVO { return _playerModel.getPlayer(id); }

public function getParticipantRating( id:String ):int { return _battleModel.getParticipantRating(id);
}

public function getBattleEntitiesByPlayer( id:String, category:String = CategoryEnum.SHIP
):Vector.<BattleEntityVO> { return _battleModel.getBattleEntitiesByPlayer(id, category); }

public function exitCombat():void
{
if (_battleModel.oldGameState == StateEvent.GAME_SECTOR)
{
var fleetID:String = null;
var sectorEvent:SectorEvent;
var sectorID:String = _battleModel.oldSector;
if (_sectorModel.focusFleetID != null)
{
var fleetVO:FleetVO = _fleetModel.getFleet(_sectorModel.focusFleetID);
if (fleetVO.sector == _battleModel.oldSector)
fleetID = _sectorModel.focusFleetID;
else if (fleetVO.sector == _sectorModel.sectorID)
{
fleetID = _sectorModel.focusFleetID;
sectorID = _sectorModel.sectorID;
}
}
if (fleetID != null)
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sectorID, fleetID);
else
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sectorID, null,
_battleModel.focusLocation.x, _battleModel.focusLocation.y);

dispatch(sectorEvent);
} else
{
var event:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(event);
}
}

override public function loadBackground(battleModel:BattleModel, useModelData:Boolean =
false):void
{
super.loadBackground(battleModel, useModelData);
//position
```

```
the view on the ships
var battleEntities:Vector.<BattleEntityVO> = _battleModel.battleEntities;
var entity:Entity;
var position:Position;
if (_battleModel.isBaseCombat)
{
for (var i:int = 0; i < battleEntities.length; i++)
{
if (battleEntities[i].category == CategoryEnum.SHIP && battleEntities[i].healthPercent > 0)
{
entity = _game.getEntity(battleEntities[i].id);
if (entity)
{
position = entity.get(Position);
_system.jumpToLocation(position.x, position.y);
}
}
}
} else if (_battleModel.timeRemaining != 0)
{
//battle has already started. find the ships in the battle and center on them
var p1:Point;
var p2:Point;
var lastOwner:String;
for (i = 0; i < battleEntities.length; i++)
{
entity = _game.getEntity(battleEntities[i]);
if (entity)
{
position = entity.get(Position);
if (!p1)
{
p1 = new Point(position.x, position.y);
lastOwner = battleEntities[i].ownerID;
} else if (battleEntities[i].ownerID != lastOwner)
p2 = new Point(position.x, position.y);
}
}
//we found two ships owned by different players
if (p1 && p2)
{
var p3:Point = Point.interpolate(p1, p2, 0.5);
_system.jumpToLocation(p3.x, p3.y);
}
//only found one ship
else if (p1)
_system.jumpToLocation(p1.x, p1.y);
}
}

private
```

```
function orderParticipants( participantA:String, participantB:String ):int
{
var playerA:PlayerVO = getPlayer(participantA);
var playerB:PlayerVO = getPlayer(participantB);

if (playerA.id == CurrentUser.id || (playerA.faction == CurrentUser.battleFaction &&
!playerA.isNPC))
return -1;
if (playerB.id == CurrentUser.id || (playerB.faction == CurrentUser.battleFaction &&
!playerB.isNPC))
return 1;
if (!playerA.isNPC && playerB.isNPC)
return -1;
if (!playerB.isNPC && playerA.isNPC)
return 1;
return 0;
}

public function isPlayerBaseOwner( playerID:String ):Boolean { return
_battleModel.baseOwnerID == playerID; }
public function isInstancedMission():Boolean {return _battleModel.isInstancedMission;}
public function isPlayerInCombat( playerID:String ):Boolean { return (_battleModel.participants)
? _battleModel.participants.indexOf(playerID) > -1 : false; }

public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getStoreItemPrototypeByName( name:String ):IPrototype
{
var proto:IPrototype = _prototypeModel.getStoreItemPrototypeByName(name);
return proto;
}

public function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number
{
return _transactionController.getBlueprintHardCurrencyCost(blueprint, partsPurchased);
}

public function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void
{
_transactionController.buyBlueprintTransaction(blueprint, partsPurchased);
}
public function completeBlueprintResearch( blueprint:BlueprintVO):void
{
_transactionController.completeBlueprintResearchTransaction(blueprint);
}

public
```

```
function purchaseReroll( battleKey:String ):void
{
_transactionController.starbasePurchaseReroll(battleKey);
}

public function purchaseDeepScan( battleKey:String ):void
{
_transactionController.starbasePurchaseDeepScan(battleKey);
}

public function addRerollFromRerollCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.add(callback);
}

public function removeRerollFromRerollCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.remove(callback);
}

public function addRerollFromScanCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.add(callback);
}

public function removeRerollFromScanCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.remove(callback);
}

public function removeRerollFromAvailable( battleID:String ):void
{
_battleModel.removeRerollByID(battleID);
}

public function getBlueprintByID( id:String ):BlueprintVO
{
return _blueprintModel.getBlueprintByID(id);
}

public function removeBlueprintByName( name:String ):void
{
_blueprintModel.removeBlueprintByName(name);
}

public function getBlueprintByName( name:String ):BlueprintVO { return
_blueprintModel.getBlueprintByName(name); }

public function getAvailableRerollById( id:String ):BattleRerollVO { return
_battleModel.getAvailableRerollByID(id); }

public
```

```
function addListenerBattleEntitiesControlledUpdated( listener:Function ):void
{
if (_system)
_system.onControlledUpdated.add(listener);
}

public function getBlueprintPrototypeByName( v:String ):IPrototype
{
return _prototypeModel.getBlueprintPrototype(v);
}

public function getResearchPrototypeByName( v:String ):IPrototype
{
return _prototypeModel.getResearchPrototypeByName(v);
}

public function getShipPrototype( v:String ):IPrototype
{
return _prototypeModel.getShipPrototype(v);
}

public function getModulePrototypeByName( v:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getShipPrototype(v);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(v);
return iproto;
}

public function getConstantPrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getConstantPrototypeByName(name);
}

public function getBEDialogueByFaction( faction:String, result:String = 'Victory'
):Vector.<IPrototype>
{
return _prototypeModel.getBEDialogueByFaction(faction, result);
}

public function getUnavailableReroll( id:String ):int
{
return _battleModel.getUnavailableRerollByID(id);
}

public function get battleRunning():Boolean { return !_battleModel.finished; }
public function get battleTimeRemaining():Number { return _battleModel.timeRemaining; }
public function get doesPlayerOwnBase():Boolean { return true; }
public function get isBaseCombat():Boolean { return _battleModel.isBaseCombat; }
public function get isFTERunning():Boolean { return _fteController.running; }
public
```

```
function get ownedBlueprints():Dictionary { return _blueprintModel.ownedBlueprints; }
public function get participants():Vector.<String> { return _battleModel.participants; }
public function get players():Dictionary { return _playerModel.getPlayers(); }
public function get showRetreat():Boolean { return isBaseCombat &&
!isPlayerBaseOwner(CurrentUser.id) && isPlayerInCombat(CurrentUser.id) &&
!getPlayer(_battleModel.baseOwnerID).isNPC; }

public function get isPVEBattle():Boolean
{
var vo:PlayerVO;
for (var i:int = 0; i < _battleModel.participants.length; i++)
{
vo = getPlayer(_battleModel.participants[i]);
if (vo.isNPC)
return true;
}
return false;
}

public function getPrototypeByName( proto:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getBuildingPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getResearchPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getShipPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getStoreItemPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(proto);
return iproto;
}

public function addListenerOnParticipantsAdded( listener:Function ):void
{
_battleModel.onParticipantsAdded.add(listener);
}

public function removeListenerOnParticipantsAdded( listener:Function ):void
{
_battleModel.onParticipantsAdded.remove(listener);
}

[Inject]
public function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set blueprintModel( v:BlueprintModel ):void { _blueprintModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public
```

```
function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }

override public function destroy():void
{
super.destroy();
_abilitySignal.removeAll();
_abilitySignal = null;
_battleModel = null;
_blueprintModel = null;
_fleetModel = null;
_playerModel = null;
_sectorModel = null;
_serverController = null;
_startSignal.removeAll();
_startSignal = null;
_system = null;
}
}
}
```

File 358: igw\com\presenter\battle\IBattlePresenter.as

```
package com.presenter.battle
{
import com.enum.CategoryEnum;
import com.model.asset.AssetVO;
import com.model.battle.BattleEntityVO;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintVO;
import com.model.fleet.FleetVO;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.IGamePresenter;

import flash.utils.Dictionary;

import org.ash.core.Entity;

public interface IBattlePresenter extends IGamePresenter
{
function getModuleAssetVo( entity:Entity, idx:Number ):AssetVO;
function
```

```
loadSmallImage( portraitName:String, callback:Function ):void;
function loadMediumImage( portraitName:String, callback:Function ):void;

function addStartListener( callback:Function ):void;
function removeStartListener( callback:Function ):void;
function onBattleStarted():void;
function onBattleEnded():void;
function sharePvPVictory():void;

function getPlayer( id:String ):PlayerVO;
function getParticipantRating( id:String ):int;
function getBattleEntitiesByPlayer( id:String, category:String = CategoryEnum.SHIP
):Vector.<BattleEntityVO>;
function exitCombat():void;
function selectOwnedShipById( shipId:String ):void;

function getSelectedFleet():FleetVO;
function getShip( shipId:String ):Entity;
function loadIcon( url:String, callback:Function ):void;
function getHealthPercentByPlayerID( id:String ):Number;
function loadMiniIconFromEntityData( type:String, callback:Function ):void;
function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number;
function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void;
function purchaseReroll( battleKey:String ):void;
function purchaseDeepScan( battleKey:String ):void;
function addRerollFromRerollCallback( callback:Function ):void;
function removeRerollFromRerollCallback( callback:Function ):void;
function addRerollFromScanCallback( callback:Function ):void;
function removeRerollFromScanCallback( callback:Function ):void;
function removeRerollFromAvailable( battleID:String ):void;

function addListenerVitalPercentUpdates( listener:Function ):void;
function removeListenerVitalPercentUpdates( listener:Function ):void;

function getConstantPrototypeValueByName( name:String ):Number;
function getStoreItemPrototypeByName( name:String ):IPrototype;
function isPlayerBaseOwner( playerID:String ):Boolean;
function isPlayerInCombat( playerID:String ):Boolean;
function getBlueprintByName( name:String ):BlueprintVO;
function getAvailableRerollById( id:String ):BattleRerollVO;
function getBlueprintByID( id:String ):BlueprintVO;
function removeBlueprintByName( name:String ):void;
function isInstancedMission():Boolean;

function getBlueprintPrototypeByName( v:String ):IPrototype;
function getResearchPrototypeByName( v:String ):IPrototype;
function getShipPrototype( v:String ):IPrototype;
function getModulePrototypeByName( v:String ):IPrototype;
function getConstantPrototypeByName( name:String ):IPrototype;
function
```

```
getBEDialogueByFaction( faction:String, result:String = 'Victory' ):Vector.<IPrototype>;
function getPrototypeByName( proto:String ):IPrototype;
function getUnavailableReroll( id:String ):int;

function addListenerBattleEntitiesControlledUpdated( listener:Function ):void;

function addListenerOnParticipantsAdded( listener:Function ):void;
function removeListenerOnParticipantsAdded( listener:Function ):void;

function get battleRunning():Boolean;
function get battleTimeRemaining():Number;
function get doesPlayerOwnBase():Boolean;
function get isBaseCombat():Boolean;
function get isFTERunning():Boolean;
function get isPVEBattle():Boolean;
function get ownedBlueprints():Dictionary;
function get participants():Vector.<String>;
function get players():Dictionary;
function get showRetreat():Boolean;
}
}

-------------------------------------
File 359: igw\com\presenter\battle\IWarfrontPresenter.as
package com.presenter.battle
{
import com.model.warfrontModel.WarfrontVO;
import com.presenter.IImperiumPresenter;

public interface IWarfrontPresenter extends IImperiumPresenter
{
function watchBattle( battle:WarfrontVO ):void;

function loadPortraitSmall( portraitName:String, callback:Function ):void;

function addUpdateListener( listener:Function ):void;
function removeUpdateListener( listener:Function ):void;

function get battles():Vector.<WarfrontVO>;
}
}

-------------------------------------
File 360: igw\com\presenter\battle\WarfrontPresenter.as
package com.presenter.battle
{
import com.event.BattleEvent;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.sector.SectorModel;
import
```

```
com.model.warfrontModel.WarfrontModel;
import com.model.warfrontModel.WarfrontVO;
import com.presenter.ImperiumPresenter;

public class WarfrontPresenter extends ImperiumPresenter implements IWarfrontPresenter
{
private var _assetModel:AssetModel;
private var _sectorModel:SectorModel;
private var _warfrontModel:WarfrontModel;

[PostConstruct]
override public function init():void
{
super.init();
}

public function watchBattle( battle:WarfrontVO ):void
{
var battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN,
battle.battleServerAddress);
dispatch(battleEvent);
}

public function loadPortraitSmall( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(AssetModel.instance.getFromCache(portraitName));
_assetModel.getFromCache('assets/' + avatarVO.smallImage, callback);
}

public function addUpdateListener( listener:Function ):void {
_warfrontModel.addUpdateListener(listener); }
public function removeUpdateListener( listener:Function ):void {
_warfrontModel.removeUpdateListener(listener); }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
public function get battles():Vector.<WarfrontVO> { return _warfrontModel.battles; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set warfrontModel( v:WarfrontModel ):void { _warfrontModel = v; }

override public function destroy():void
{
super.destroy();
_assetModel = null;
_sectorModel = null;
_warfrontModel = null;
}
}
}
```

File 361: igw\com\presenter\preload\IPreloadPresenter.as

```
package com.presenter.preload
{
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;

import org.osflash.signals.Signal;

public interface IPreloadPresenter extends IImperiumPresenter
{
function trackPlayerProgress( id:int ):void;
function beginLoad():void;

function transitionToLoad():void;
function sendCharacterToServer( factionPrototype:String, racePrototype:String ):void;
function getRacePrototypesByFaction( faction:String, race:String ):Vector.<IPrototype>;
function getRacePrototypeByName( race:String ):IPrototype;
function getFirstNameOptions( race:String, gender:String ):Vector.<IPrototype>;
function getLastNameOptions( race:String, gender:String ):Vector.<IPrototype>;
function getAudioProtos():Vector.<IPrototype>;
function getEntityData( type:String ):AssetVO;

function loadPortraitIcon( portraitName:String, callback:Function ):void;
function loadPortraitLarge( portraitName:String, callback:Function ):void;

function addLoadCompleteListener( callback:Function ):void;
function removeLoadCompleteListener( callback:Function ):void;

function get beginSignal():Signal;
function get completeSignal():Signal;
function get progressSignal():Signal;
}
}
```

File 362: igw\com\presenter\preload\PreloadPresenter.as

```
package com.presenter.preload
{
import com.Application;
import com.controller.ServerController;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.ServerEvent;
import com.event.StateEvent;
import com.event.TransitionEvent;
import
```

```
com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;
import com.service.loading.ILoadService;
import com.service.loading.LoadService;
import com.service.server.outgoing.proxy.ProxyTutorialStepCompletedMessage;
import com.service.server.outgoing.universe.UniverseCreateCharacterRequest;

import flash.events.Event;
import flash.events.ProgressEvent;
import flash.text.Font;
import flash.utils.getDefinitionByName;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.osflash.signals.Signal;
import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;

public class PreloadPresenter extends ImperiumPresenter implements IPreloadPresenter
{
public static var complete:Boolean = false;

private static var _masterLoaded:Boolean = false;

private const _logger:ILogger = getLogger('PreloadPresenter');

private var _assetModel:AssetModel;
private var _beginSignal:Signal;
private var _completeSignal:Signal;
private var _loadCompleteSignal:Signal;
private var _loadService:ILoadService;
private var _progressSignal:Signal;
private var _prototypeModel:PrototypeModel;
private var _serverController:ServerController;
private var _starbaseModel:StarbaseModel;

[PostConstruct]
override public function init():void
{
super.init();
_beginSignal = new Signal(int);
_completeSignal = new Signal();
_loadCompleteSignal = new Signal();
_progressSignal = new Signal(int, int);
}

public
```

```
function beginLoad():void
{
_eventMap.mapListener(_eventDispatcher, ProgressEvent.PROGRESS, onProgress,
ProgressEvent, false, 0, true);
_eventMap.mapListener(_eventDispatcher, LoadService.ALL_COMPLETE, onLoadComplete,
Event, false, 0, true);

if (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER)
_loadService.lazyLoad('LoadMaster.xml');
else
_loadService.lazyLoad('LoadMobileMaster.xml');
}

public function trackPlayerProgress( id:int ):void
{
var msg:ProxyTutorialStepCompletedMessage =
ProxyTutorialStepCompletedMessage(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_TUTORIAL_STEP_COMPLETED));
msg.stepId = id;
msg.kabamNaid = CurrentUser.naid;
_serverController.send(msg);
}

private function preloadFromMaster():void
{
var xml:XMLList;
if (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER)
xml = XML(_assetModel.getFromCache('LoadMaster.xml')).children();
else
xml = XML(_assetModel.getFromCache('LoadMobileMaster.xml')).children();

var nodes:XMLList;
for (var i:int = 0; i < xml.length(); i++)
{
nodes = xml[i].children();
for (var j:int = 0; j < nodes.length(); j++)
{
_loadService.lazyLoad(nodes[j]);
}
}

_beginSignal.dispatch(xml.length());
}

private function onProgress( e:ProgressEvent ):void
{
if (_masterLoaded)
_progressSignal.dispatch(e.bytesLoaded, e.bytesTotal);
}

private
```

```
function onLoadComplete( e:Event ):void
{
if (_masterLoaded)
{
var xml:XMLList = (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER) ?
XML(_assetModel.getFromCache('LoadMaster.xml')).children() :
XML(_assetModel.getFromCache('LoadMobileMaster.xml')).
children();
var name:String;

if (DeviceMetrics.PLATFORM == PlatformEnum.MOBILE)
{
Font.registerFont(Class(getDefinitionByName('Agency')));
Font.registerFont(Class(getDefinitionByName('AgencyBold')));
Font.registerFont(Class(getDefinitionByName('OpenSansBold')));
Font.registerFont(Class(getDefinitionByName('OpenSansRegular')));
} else
{
for (var i:int = 0; i < xml.length(); i++)
{
name = xml[i].name();
switch (name)
{
case 'fonts':
registerFonts(xml[i]);
break;
}
}
}

//remove the xml file from memory
if (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER)
_assetModel.removeFromCache('LoadMaster.xml');
else
_assetModel.removeFromCache('LoadMobileMaster.xml');

_progressSignal.removeAll();
_loadCompleteSignal.dispatch();

_eventMap.unmapListener(_eventDispatcher, ProgressEvent.PROGRESS, onProgress,
ProgressEvent);
_eventMap.unmapListener(_eventDispatcher, LoadService.ALL_COMPLETE, onLoadComplete,
Event);
} else if (!_masterLoaded)
{
_masterLoaded = true;
preloadFromMaster();
}
}

public
```

```
function transitionToLoad():void
{
_loadService.reset();
complete = true;
var stateEvent:StateEvent;
if (Application.CONNECTION_STATE == ServerEvent.NEED_CHARACTER_CREATE)
stateEvent = new StateEvent(StateEvent.CREATE_CHARACTER);
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
transitionEvent.addEvents(stateEvent, new StateEvent(StateEvent.PRELOAD_COMPLETE));
dispatch(transitionEvent);

if (Application.CONNECTION_STATE == ServerEvent.NOT_CONNECTED)
dispatch(new ServerEvent(ServerEvent.CONNECT_TO_PROXY));
}

public function sendCharacterToServer( factionPrototype:String, racePrototype:String ):void
{
var transitionEvent:TransitionEvent = new
TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
transitionEvent.addEvents(null, null);
dispatch(transitionEvent);

var createCharacterRequest:UniverseCreateCharacterRequest =
UniverseCreateCharacterRequest(_serverController.getRequest(ProtocolEnum.UNIVERSE_CLIENT,
RequestEnum.UNIVERSE_CHARACTER_CREATION_REQUEST));
createCharacterRequest.factionPrototype = factionPrototype;
createCharacterRequest.racePrototype = racePrototype;
createCharacterRequest.name = CurrentUser.name;
_serverController.send(createCharacterRequest);
}

public function getRacePrototypesByFaction( faction:String, race:String ):Vector.<IPrototype>
{
return _prototypeModel.getRacePrototypesByFaction(faction, race);
}

public function getRacePrototypeByName( race:String ):IPrototype
{
return _prototypeModel.getRacePrototypeByName(race);
}

public function getFirstNameOptions( race:String, gender:String ):Vector.<IPrototype>
{
return _prototypeModel.getFirstNameOptions(race, gender);
}

public function getLastNameOptions( race:String, gender:String ):Vector.<IPrototype>
{
return _prototypeModel.getLastNameOptions(race, gender);
}
```

```
public function getAudioProtos():Vector.<IPrototype>
{
return _assetModel.getAudioProtos();
}

public function getEntityData( type:String ):AssetVO
{
return _assetModel.getEntityData(type);
}

public function addLoadCompleteListener( callback:Function ):void {
_loadCompleteSignal.add(callback); }
public function removeLoadCompleteListener( callback:Function ):void {
_loadCompleteSignal.remove(callback); }

/**
* Take the fonts that were loaded in and register them for use
* @param node xml list of the fonts
*/
private function registerFonts( node:XML ):void
{
var fontNames:Array = node.@names.split(",");
for (var i:int = 0; i < fontNames.length; i++)
{
Font.registerFont(Class(getDefinitionByName(fontNames[i])));
}
}

public function loadPortraitIcon( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(_assetModel.getFromCache(portraitName));
_assetModel.getFromCache('assets/' + avatarVO.iconImage, callback);
}

public function loadPortraitLarge( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(_assetModel.getFromCache(portraitName));
_assetModel.getFromCache('assets/' + avatarVO.largeImage, callback);
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }

public function get beginSignal():Signal { return _beginSignal; }
public function get completeSignal():Signal { return _completeSignal; }
[Inject]
public function set loadService( v:ILoadService ):void { _loadService = v; }
public function get progressSignal():Signal { return _progressSignal; }
[Inject]
```

```
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }

override public function destroy():void
{
super.destroy();
_loadService = null;
_serverController = null;
_beginSignal.removeAll();
_beginSignal = null;
_completeSignal.removeAll();
_completeSignal = null;
_loadCompleteSignal.removeAll();
_loadCompleteSignal = null;
_progressSignal.removeAll();
_progressSignal = null;
_prototypeModel = null;
_starbaseModel = null;
}
}
}

------------------------------------
File 363: igw\com\presenter\sector\IMiniMapPresenter.as
package com.presenter.sector
{
import com.game.entity.components.shared.Position;
import com.model.mission.MissionVO;
import com.model.player.PlayerVO;
import com.presenter.IImperiumPresenter;

import flash.geom.Point;

import org.ash.core.Entity;
import org.osflash.signals.Signal;

public interface IMiniMapPresenter extends IImperiumPresenter
{
function get addToMiniMapSignal():Signal;
function get clearMiniMapSignal():Signal;
function get fteRunning():Boolean;
function get removeFromMiniMapSignal():Signal;
function get scrollMiniMapSignal():Signal;

function enterStarbase():void;
function isInInstancedMission():Boolean;
function
```

```
enterInstancedMission():void;
function updateScale():void;
function mouseDown():void;
function mouseUp():void;
function mouseMove( xDelta:Number, yDelta:Number ):void;
function mouseWheel( delta:Number ):void;

function getEntity( id:String ):Entity;
function showSector():void;
function retreat():void;
function findBase( userName:String ):void;
function moveToMissionTarget():String;

function addListenerOnCoordsUpdate( listener:Function ):void;
function removeListenerOnCoordsUpdate( listener:Function ):void;

function get currentMission():MissionVO;

function addListenerToUpdateMission( listener:Function ):void;
function removeListenerToUpdateMission( listener:Function ):void;

function get isMissionBattle():Boolean;
function get mapWidth():Number;
function set mapWidth( v:Number ):void;
function get miniMapWidth():Number;
function set miniMapWidth( v:Number ):void;
function get sectorName():String;
function get sectorEnum():String;
function get showRetreat():Boolean;
function get zoom():Number;
function get zoomPercent():Number;
function set zoomPercent( v:Number ):void;
function getPlayer( id:String ):PlayerVO;
function getConstantPrototypeByName( v:String ):*;
function get focusedFleetRating():int;

function addSelectionChangeListener( listener:Function ):void;
function removeSelectionChangeListener( listener:Function ):void;

function getIconPosition( width:Number, height:Number, entityPosition:Position ):Point;
}
}

------------------------------------
File 364: igw\com\presenter\sector\ISectorPresenter.as
package com.presenter.sector
{
import com.model.fleet.FleetVO;
import com.model.mission.MissionVO;
import com.model.player.PlayerVO;
import
```

```
com.model.sector.SectorVO;
import com.model.starbase.BaseVO;
import com.presenter.shared.IGamePresenter;

import org.ash.core.Entity;
import com.model.prototype.IPrototype

public interface ISectorPresenter extends IGamePresenter
{
function onInteractionWithSectorEntity( x:int, y:int, entity:Entity, selectedEntity:Entity ):void;
function onBattle():void;
function joinBattle( battleServerAddress:String ):void;
function selectFleet( fleetID:String, gotoLocation:Boolean = true, canEnterBattle:Boolean = true,
canChangeSector:Boolean = true ):Boolean;
function attackEntity( entity:Entity, ignoreVerification:Boolean = false ):void;
function tackleEntity( entity:Entity, ignoreVerification:Boolean = false ):void;
function recallFleet( entity:Entity, targetTransgate:Entity = null ):void;
function defendBase( entity:Entity, targetBase:Entity ):void
function watchBattle( entity:Entity ):void;
function enterStarbase( entity:Entity ):void;
function travelViaTransgate( sector:String, entity:Entity, target:Entity ):void;
function relocateToTransgate( sector:String, targetTransgate:String ):void;
function travelToWaypoint(entity:Entity, target:Entity ):void;
function lootDerelictFleet( entity:Entity ):void;
function jumpToLocation( x:Number, y:Number ):void;
function getFleetVO( id:String ):FleetVO;
function loadIcon( url:String, callback:Function ):void
function loadMiniIconFromEntityData( type:String, callback:Function ):void;
function removeTargetSelection():void;
function getTransgateDestinations():Vector.<SectorVO>;
function getPrivateDestinations():Vector.<SectorVO>;
function getPlayer( id:String ):PlayerVO;
function getConstantPrototypeByName( v:String ):*;

function getTransgateCustomDestinationPrototype( key:String ):IPrototype;
function getTransgateCustomDestinationGroupByCustomDestinationGroup( group:String
):Vector.<IPrototype>;

function addInteractListener( listener:Function ):void;
function removeInteractListener( listener:Function ):void;

function addListenerOnCoordsUpdate( listener:Function ):void;
function removeListenerOnCoordsUpdate( listener:Function ):void;

function addBattleListener( listener:Function ):void;
function removeBattleListener( listener:Function ):void;

function addListenerForFleetUpdate( listener:Function ):void;
function removeListenerForFleetUpdate( listener:Function ):void;

function
```

```
addNotificationListener( listener:Function ):void;
function removeNotificationListener( listener:Function ):void;

function addSelectionChangeListener( listener:Function ):void;
function addOnGenericAllianceMessageRecievedListener( callback:Function ):void;
function removeOnGenericAllianceMessageRecievedListener( callback:Function ):void;

function getBase( id:String ):BaseVO;
function getFleets():Vector.<FleetVO>;

function get sectorName():String;
function get sectorEnum():String;
function get currentMission():MissionVO;
function get neighborhood():int;
function get focusFleetID():String;
function get selectedEntity():Entity;
function get selectedEnemy():Entity;
function get sectorID():String;
}
}

-------------------------------------
File 365: igw\com\presenter\sector\MiniMapPresenter.as
package com.presenter.sector
{
import com.Application;
import com.controller.ServerController;
import com.enum.TypeEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.SectorEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.nodes.shared.grid.GridNode;
import com.game.entity.systems.interact.BattleInteractSystem;
import com.game.entity.systems.interact.InteractSystem;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.shared.grid.GridSystem;
import com.model.battle.BattleModel;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.mission.MissionModel;
import com.model.mission.MissionVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.PrototypeModel;
import
```

```
com.model.scene.SceneModel;
import com.model.sector.SectorModel;
import com.model.starbase.BaseVO;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;
import com.service.server.outgoing.battle.BattleRetreatRequest;

import flash.geom.Point;
import flash.geom.Rectangle;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.osflash.signals.Signal;

public class MiniMapPresenter extends ImperiumPresenter implements IMiniMapPresenter
{
private const ZOOM_MIN:Number = 0.5;
private const ZOOM_MAX:Number = 2.5;
private const ZOOM_RANGE:Number = ZOOM_MAX - ZOOM_MIN;

private var _battleModel:BattleModel;
private var _combinedFactor:Number;
private var _dragStart:Point;
private var _fleetModel:FleetModel;
private var _game:Game;
private var _missionModel:MissionModel;
private var _scale:Number;
private var _sceneModel:SceneModel;
private var _sectorModel:SectorModel;
private var _serverController:ServerController;
private var _starbaseModel:StarbaseModel;
private var _playerModel:PlayerModel;
private var _prototypeModel:PrototypeModel;
private var _zoom:Number = 1.0;

/** The actual world coord size of the map the minimap represents. */
private var _mapWidth:Number = 0.0;

/** The size of the minimap's viewport; used to set the scale factor. */
private var _miniMapWidth:Number = 0.0;

private var _interactSystem:InteractSystem;

private var _addToMiniMapSignal:Signal;
private var _clearMiniMapSignal:Signal;
private var _removeFromMiniMapSignal:Signal;
private var _scrollMiniMapSignal:Signal;

private var _launchFleetForMission:String = 'CodeString.Toast.LaunchFleetForMission';
private var _returnFleetToHomeSectorForMission:String =
'CodeString.Toast.ReturnFleetToHomeSectorForMission';
```

```
[PostConstruct]
override public function init():void
{
super.init();
_addToMiniMapSignal = new Signal(Entity, Rectangle);
_removeFromMiniMapSignal = new Signal(Entity);
_scrollMiniMapSignal = new Signal();
_clearMiniMapSignal = new Signal();
}

public function updateScale():void
{
_scale = _mapWidth > 0 ? _miniMapWidth / _mapWidth : 0;
_combinedFactor = _scale * _zoom;
}

public function getIconPosition( width:Number, height:Number, entityPosition:Position ):Point
{
var result:Point = new Point();
var xCenter:Number = width >> 1;
result.x = xCenter + (entityPosition.x - _sceneModel.focus.x) * _combinedFactor;

var yCenter:Number = height >> 1;
result.y = yCenter + (entityPosition.y - _sceneModel.focus.y) * _combinedFactor;

return result;
}

override protected function onStateChange( e:StateEvent ):void
{
_interactSystem = null;
super.onStateChange(e);
}

public function mouseDown():void
{
_dragStart = _sceneModel.focus.clone();
interactSystem.followEntity = null;
}

public function mouseUp():void
{

}

public function mouseMove( xDelta:Number, yDelta:Number ):void
{
if (interactSystem)
interactSystem.jumpToLocation(_dragStart.x
```

```
- (xDelta / _combinedFactor), _dragStart.y - (yDelta / _combinedFactor));
}

public function mouseWheel( delta:Number ):void
{
zoom += delta / 30;
}



public function enterStarbase():void
{
var event:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(event);
}

public function isInInstancedMission():Boolean
{
return (_starbaseModel.homeBase.instancedMissionAddress != null);
}
public function enterInstancedMission():void
{
var event:StarbaseEvent = new
StarbaseEvent(StarbaseEvent.ENTER_INSTANCED_MISSION);
dispatch(event);
}

public function showSector():void
{
if (_fteController.running)
return;
var sectorEvent:SectorEvent;
if ((Application.STATE == StateEvent.GAME_BATTLE_INIT || Application.STATE ==
StateEvent.GAME_BATTLE) && (!_battleModel.isBaseCombat || _battleModel.baseOwnerID !=
CurrentUser.id))
{
if (_battleModel.oldGameState == StateEvent.GAME_SECTOR)
{
var fleetID:String = null;
var sectorID:String = _battleModel.oldSector;
if (_sectorModel.focusFleetID != null)
{
var fleetVO:FleetVO = _fleetModel.getFleet(_sectorModel.focusFleetID);
if (fleetVO.sector == _battleModel.oldSector)
fleetID = _sectorModel.focusFleetID;
else if (fleetVO.sector == _sectorModel.sectorID)
{
fleetID = _sectorModel.focusFleetID;
sectorID = _sectorModel.sectorID;
}
}
```

```
if (fleetID != null)
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sectorID, fleetID);
else
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sectorID, null,
_battleModel.focusLocation.x, _battleModel.focusLocation.y);

sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, _battleModel.oldSector, null,
_battleModel.focusLocation.x, _battleModel.focusLocation.y);
dispatch(sectorEvent);
} else
{
var event:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(event);
}
} else
{
_sectorModel.viewBase = true;
_sectorModel.focusFleetID = null;
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR);
dispatch(sectorEvent);
}
}

public function retreat():void
{
var request:BattleRetreatRequest =
BattleRetreatRequest(_serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_RETREAT));
_serverController.send(request);
}

public function findBase( userName:String ):void
{
var baseVO:BaseVO = (userName == CurrentUser.id) ? _starbaseModel.currentBase : null;
if (baseVO)
{
if (baseVO.sectorID == _sectorModel.sectorID)
interactSystem.jumpToLocation(baseVO.sectorLocationX, baseVO.sectorLocationY);
else
{
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR,
baseVO.sectorID);
_sectorModel.viewBase = true;
dispatch(sectorEvent);
}
}
}

private function findClosestTransgate( target:Entity ):Entity
{
```

```
var entity:Entity;
var bestDistSq:Number = 0;
var targetPos:Position = target.get(Position);

// This will only work in sector mode, because it's the only place transgates are found, naturally.
var detail:Detail;
var pos:Position;
var distSq:Number;
var gridSystem:GridSystem = GridSystem(_game.getSystem(GridSystem));
for (var node:GridNode = gridSystem.nodes.head; node; node = node.next)
{
detail = node.entity.get(Detail);
switch (detail.type)
{
case TypeEnum.TRANSGATE_IGA:
case TypeEnum.TRANSGATE_SOVEREIGNTY:
case TypeEnum.TRANSGATE_TYRANNAR:
break;
default:
continue;
}

pos = node.entity.get(Position);
distSq = (pos.x - targetPos.x) * (pos.x - targetPos.x) + (pos.y - targetPos.y) * (pos.y -
targetPos.y);
if (bestDistSq <= 0 || distSq < bestDistSq)
{
entity = node.entity;
bestDistSq = distSq;
}
}

return entity;
}

public function moveToMissionTarget():String
{
var mission:MissionVO = _missionModel.currentMission;
//send the player back to their base if they're doing a non-kill mission
if (mission.progressEvent != "Kill")
{
if (Application.STATE != StateEvent.GAME_STARBASE)
{
var starbaseEvent:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(starbaseEvent);
}
return null;
}

var
```

```
fleet:FleetVO;
var fleets:Vector.<FleetVO> = _fleetModel.fleets;
var activeFleet:FleetVO;
var hasFleetLaunched:Boolean = false;
for (var i:int = 0; i < fleets.length; i++)
{
fleet = fleets[i];
if (fleet.sector != "")
{
hasFleetLaunched = true;
}
}
if(!hasFleetLaunched)
{
return _launchFleetForMission;
}

var activeFleetId:String = _sectorModel.focusFleetID;
activeFleet = _fleetModel.getFleet(activeFleetId);

//TODO: handle the case when no fleet is active but still launched (e.g., recently selected fleet is
not in the current sector)
if(!activeFleet)
return null;

var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system)
{
//are we in the same sector as the mission entity?
if (mission.sector == _sectorModel.sectorID)
{
var missionEntities:Vector.<Entity> = system.missionEntities;
if (missionEntities.length > 0 && activeFleet)
{
var entity:Entity = missionEntities[0];
var position:Position = entity.get(Position);
system.moveToLocation(position.x, position.y, 1.3);
return null;
}
} else
{
//we're not in the same sector so point to a nearby transgate we can travel through
if (activeFleet)
{
var fleetEntity:Entity = _game.getEntity(activeFleetId);
if (fleetEntity)
{
var closestTransgate:Entity = findClosestTransgate(fleetEntity);
if (closestTransgate)
{
```

```
var transgatePos:Position = closestTransgate.get(Position);
system.moveToLocation(transgatePos.x, transgatePos.y, 1.3);
}
}
return null;
}
}
return _launchFleetForMission;
}

var event:SectorEvent;
if (activeFleet)
event = new SectorEvent(SectorEvent.CHANGE_SECTOR, activeFleet.sector, activeFleet.id);
if (event)
{
dispatch(event);
return null;
}

if (Application.STATE == StateEvent.GAME_SECTOR)
{
if (hasFleetLaunched &&
mission.sector != _sectorModel.sectorID &&
_starbaseModel.homeBase.sectorID != _sectorModel.sectorID)
{
return _returnFleetToHomeSectorForMission;
}
}

return _launchFleetForMission;
}

public function getConstantPrototypeByName( v:String ):*
{
return _prototypeModel.getConstantPrototypeValueByName(v);
}

public function getEntity( id:String ):Entity { return _game.getEntity(id); }

public function addListenerOnCoordsUpdate( listener:Function ):void {
SectorInteractSystem(_game.getSystem(SectorInteractSystem)).onCoordsUpdate.add(listener);
}
public function removeListenerOnCoordsUpdate( listener:Function ):void {
SectorInteractSystem(_game.getSystem(SectorInteractSystem)).onCoordsUpdate.remove(listener);
}

public function get addToMiniMapSignal():Signal { return _addToMiniMapSignal; }
public function get clearMiniMapSignal():Signal { return _clearMiniMapSignal; }
public function get fteRunning():Boolean { return _fteController.running; }

public
```

```
function get mapWidth():Number { return _mapWidth; }
public function set mapWidth( value:Number ):void { _mapWidth = value; updateScale(); }

public function get miniMapWidth():Number { return _miniMapWidth; }
public function set miniMapWidth( value:Number ):void { _miniMapWidth = value; updateScale();
}

public function get removeFromMiniMapSignal():Signal { return _removeFromMiniMapSignal; }
public function get scrollMiniMapSignal():Signal { return _scrollMiniMapSignal; }

public function get sectorName():String { return _sectorModel.sectorName; }
public function get sectorEnum():String { return _sectorModel.sectorEnum; }

public function get interactSystem():InteractSystem
{
if (_interactSystem)
return _interactSystem;
switch (Application.STATE)
{
case StateEvent.GAME_STARBASE:
_interactSystem = StarbaseInteractSystem(_game.getSystem(StarbaseInteractSystem));
break;
case StateEvent.GAME_BATTLE:
_interactSystem = BattleInteractSystem(_game.getSystem(BattleInteractSystem));
break;
case StateEvent.GAME_SECTOR:
_interactSystem = SectorInteractSystem(_game.getSystem(SectorInteractSystem));
break;
default:
_interactSystem = null;
}
return _interactSystem;
}

public function get isMissionBattle():Boolean { return !_missionModel.currentMission.isFTE &&
_battleModel.missionID == _missionModel.currentMission.id; }

public function get showRetreat():Boolean
{
if(_battleModel.isInstancedMission)
return true;

var baseOwner:PlayerVO = getPlayer(_battleModel.baseOwnerID);
return _battleModel.isBaseCombat && _battleModel.baseOwnerID != CurrentUser.id &&
!baseOwner.isNPC && (_battleModel.participants.indexOf(CurrentUser.id) > -1);
}

public function get zoom():Number { return _zoom; }
public function set zoom( v:Number ):void
{
v
```

```
= v < ZOOM_MIN ? ZOOM_MIN : v;
v = v > ZOOM_MAX ? ZOOM_MAX : v;

_zoom = v;
updateScale();
}

public function get focusedFleetRating():int
{
if (_sectorModel.focusFleetID)
return _fleetModel.getFleet(_sectorModel.focusFleetID).level

return 0;
}

public function get zoomPercent():Number { return (_zoom - ZOOM_MIN) / ZOOM_RANGE; }
public function set zoomPercent( v:Number ):void { zoom = ZOOM_MIN + (ZOOM_RANGE * v);
}

public function getPlayer( id:String ):PlayerVO { return _playerModel.getPlayer(id); }

public function addSelectionChangeListener( listener:Function ):void {
_sectorModel.addSelectedFleetIDChangedListener(listener); }
public function removeSelectionChangeListener( listener:Function ):void {
_sectorModel.removeSelectedFleetIDChangedListener(listener); }

public function get currentMission():MissionVO { return _missionModel.currentMission; }

public function addListenerToUpdateMission( listener:Function ):void {
_missionModel.addListenerToUpdateMission(listener); }
public function removeListenerToUpdateMission( listener:Function ):void {
_missionModel.removeListenerToUpdateMission(listener); }

[Inject]
public function set battleModel( v:BattleModel ):void { _battleModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set missionModel( v:MissionModel ):void { _missionModel = v; }
[Inject]
public function set sceneModel( v:SceneModel ):void { _sceneModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public
```

```
function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

}
}

-----------------------------------
File 366: igw\com\presenter\sector\SectorPresenter.as
package com.presenter.sector
{
import com.Application;
import com.controller.ServerController;
import com.controller.transaction.TransactionController;
import com.enum.TypeEnum;
import com.enum.server.OrderEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.BattleEvent;
import com.event.MissionEvent;
import com.event.SectorEvent;
import com.event.StarbaseEvent;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.sector.Mission;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.alliance.AllianceModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.mission.MissionModel;
import com.model.mission.MissionVO;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.model.sector.SectorVO;
import com.model.starbase.BaseVO;
import com.model.starbase.StarbaseModel;
import com.model.battle.BattleModel;
import com.presenter.shared.GamePresenter;
import com.service.server.outgoing.sector.SectorOrderRequest;
import com.service.server.outgoing.sector.SectorSetViewLocationRequest;

import com.service.ExternalInterfaceAPI;
import com.event.ServerEvent;
import com.model.prototype.IPrototype


import
```

```
org.ash.core.Entity;
import org.osflash.signals.Signal;
import org.robotlegs.extensions.presenter.impl.Presenter;

public class SectorPresenter extends GamePresenter implements ISectorPresenter
{
private var _destinations:Vector.<SectorVO>;
private var _fleetModel:FleetModel;
private var _missionModel:MissionModel;
private var _playerModel:PlayerModel;
private var _onBattleSignal:Signal;
private var _onInteractSignal:Signal;
private var _onNotificationSignal:Signal;
private var _sectorModel:SectorModel;
private var _serverController:ServerController;
private var _transactionController:TransactionController;
private var _starbaseModel:StarbaseModel;
private var _system:SectorInteractSystem;
private var _allianceModel:AllianceModel;

[PostConstruct]
override public function init():void
{
super.init();
_destinations = new Vector.<SectorVO>;
_onBattleSignal = new Signal(Entity, Boolean, String);
_onInteractSignal = new Signal(int, int, Entity, Entity);
_onNotificationSignal = new Signal(String, Entity);
_system = SectorInteractSystem(_game.getSystem(SectorInteractSystem));
_system.presenter = this;
}

public function onInteractionWithSectorEntity( x:int, y:int, entity:Entity, selectedEntity:Entity
):void
{
_onInteractSignal.dispatch(x, y, entity, selectedEntity);
}

public function onBattle():void
{
var entity:Entity = _system.selected;
if (entity)
{
var fleetVO:FleetVO = _fleetModel.getFleet(entity.id);
if (fleetVO)
_onBattleSignal.dispatch(entity, fleetVO.inBattle, fleetVO.battleServerAddress);
}
}

public function joinBattle( battleServerAddress:String ):void
{
```

```
var event:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN, battleServerAddress);
dispatch(event);
}

public function attackEntity( entity:Entity, ignoreVerification:Boolean = false ):void
{
//attack the target
if (_system.selected && entity)
{
var detail:Detail = entity.get(Detail);
if(detail == null)
return;

var showMission:Boolean = false;
//if this is a mission entity we only want to attack after showing the mission intro
if (entity.has(Mission))
{
var mission:MissionVO = _missionModel.currentMission;
if (!mission.accepted)
{
showMission = true;
var missionEvent:MissionEvent = new MissionEvent(MissionEvent.MISSION_GREETING);
dispatch(missionEvent);
}
}
var attack:Boolean = (ignoreVerification) ? true : false;
if (!ignoreVerification)
{
attack = true;
if (detail.type == TypeEnum.STARBASE_SECTOR_IGA || detail.type ==
TypeEnum.STARBASE_SECTOR_SOVEREIGNTY || detail.type ==
TypeEnum.STARBASE_SECTOR_TYRANNAR)
{
var selectedEntityPlayer:PlayerVO = getPlayer(detail.ownerID);
if(selectedEntityPlayer)
{
if (!selectedEntityPlayer.isNPC && _starbaseModel.currentBase.bubbleTimeRemaining > 0)
{
_onNotificationSignal.dispatch('bubble', entity);
attack = false;
}

if (attack && Application.NETWORK == Application.NETWORK_GUEST &&
!selectedEntityPlayer.isNPC)
{
attack = false;
ExternalInterfaceAPI.logConsole("Guest Attack Restriction");
var serverEvent:ServerEvent
serverEvent = new ServerEvent(ServerEvent.GUEST_RESTRICTION);
_eventDispatcher.dispatchEvent(serverEvent);
```

```
}
}
}
}
if (!showMission && attack)
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = _system.selected.id;
order.orderType = OrderEnum.ATTACK;
order.targetId = entity.id;
_serverController.send(order);
}
}
}

public function tackleEntity( entity:Entity, ignoreVerification:Boolean = false ):void
{
//attack the target
if (_system.selected && entity)
{
var detail:Detail = entity.get(Detail);
var showMission:Boolean = false;
//if this is a mission entity we only want to attack after showing the mission intro
if (entity.has(Mission))
{
var mission:MissionVO = _missionModel.currentMission;
if (!mission.accepted)
{
showMission = true;
var missionEvent:MissionEvent = new MissionEvent(MissionEvent.MISSION_GREETING);
dispatch(missionEvent);
}
}
var attack:Boolean = (ignoreVerification) ? true : false;
if (!ignoreVerification)
{
attack = true;
if (detail.type == TypeEnum.STARBASE_SECTOR_IGA || detail.type ==
TypeEnum.STARBASE_SECTOR_SOVEREIGNTY || detail.type ==
TypeEnum.STARBASE_SECTOR_TYRANNAR)
{
if (_starbaseModel.currentBase.bubbleTimeRemaining > 0)
{
_onNotificationSignal.dispatch('bubble', entity);
attack = false;
}
}
}
```

```
if (!showMission && attack)
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = _system.selected.id;
order.orderType = OrderEnum.TACKLE;
order.targetId = entity.id;
_serverController.send(order);
}
}
}

public function selectFleet( fleetID:String, gotoLocation:Boolean = true, canEnterBattle:Boolean
= true, canChangeSector:Boolean = true ):Boolean
{
var fleetVO:FleetVO = getFleetVO(fleetID);
if (fleetVO)
{
_sectorModel.focusFleetID = fleetVO.id;
if (!_fteController.running && fleetVO.inBattle && canEnterBattle)
{
var battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN,
fleetVO.battleServerAddress);
dispatch(battleEvent);
} else if (fleetVO.sector == _sectorModel.sectorID)
{
var entity:Entity = _game.getEntity(fleetID);
if (entity)
{
_system.selectEntity(entity, gotoLocation);
if (gotoLocation)
{
var position:Position = entity.get(Position);
jumpToLocation(position.x, position.y);
}
return true;
}
} else if (fleetVO.sector != "" && gotoLocation && canChangeSector)
{
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR,
fleetVO.sector, fleetID);
dispatch(sectorEvent);
return true;
}
}
return false;
}

public
```

```
function recallFleet( entity:Entity, targetTransgate:Entity = null ):void
{
if (entity && entity.id != null)
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = entity.id;
order.orderType = OrderEnum.RECALL;
order.targetLocationX = order.targetLocationY = 0;
order.targetId = (targetTransgate) ? targetTransgate.id : "";
_serverController.send(order);
}
}

public function defendBase( entity:Entity, targetBase:Entity ):void
{
if (entity && entity.id != null && targetBase && targetBase.id != null)
{
var fleet:FleetVO = _fleetModel.getFleet(entity.id);
if (fleet)
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = entity.id;
order.orderType = OrderEnum.DEFEND;
order.targetId = targetBase.id;
_serverController.send(order);
}
}
}

public function watchBattle( entity:Entity ):void
{
var battleServerAddress:String = Attack(entity.get(Attack)).battleServerAddress;
if (battleServerAddress)
joinBattle(battleServerAddress);
}

public function travelViaTransgate( sector:String, entity:Entity, target:Entity ):void
{
if (Application.NETWORK == Application.NETWORK_GUEST)
{
ExternalInterfaceAPI.logConsole("Guest Travel Restriction");
var serverEvent:ServerEvent
serverEvent = new ServerEvent(ServerEvent.GUEST_RESTRICTION);
_eventDispatcher.dispatchEvent(serverEvent);
}
else
{
```

```
if (entity && entity.id != null)
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = entity.id;
order.orderType = OrderEnum.TRANS_GATE_TRAVEL;
order.targetId = target.id;
order.targetLocationX = order.targetLocationY = 0;
order.destinationSector = sector;
_serverController.send(order);
}
}
}
public function relocateToTransgate( sector:String, targetTransgate:String ):void
{
_transactionController.starbaseRelocateStarbaseToTransgate(sector, targetTransgate);
}

public function travelToWaypoint(entity:Entity, target:Entity ):void
{
if (entity && entity.id != null)
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = entity.id;
order.orderType = OrderEnum.WAYPOINT_TRAVEL;
order.targetId = target.id;
order.targetLocationX = order.targetLocationY = 0;
_serverController.send(order);
}
}

public function lootDerelictFleet( entity:Entity ):void
{
if (!_system.selected || _system.selected.id == null)
return;

var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = _system.selected.id;
order.orderType = OrderEnum.SALVAGE;
order.targetId = entity.id;
_serverController.send(order);
}

public function enterStarbase( entity:Entity ):void
{
```

```
var event:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE, (entity != null) ?
entity.id : null);
dispatch(event);
}
public function enterInstancedMission():void
{
var event:StarbaseEvent = new
StarbaseEvent(StarbaseEvent.ENTER_INSTANCED_MISSION);
dispatch(event);
}

public function getFleetVO( id:String ):FleetVO
{
return _fleetModel.getFleet(id);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function loadMiniIconFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
var icon:String = _currentAssetVO.iconImage;
loadIcon(icon, callback);
}

public function removeTargetSelection():void
{
_system.removeTargetSelection();
}

override public function confirmReady():void
{
super.confirmReady();

//get the entities in our view area
var setView:SectorSetViewLocationRequest =
SectorSetViewLocationRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_SET_VIEW_LOCATION));
setView.x = _sceneModel.focus.x;
setView.y = _sceneModel.focus.y;
_serverController.send(setView);
}

override public function loadBackground(battleModel:BattleModel, useModelData:Boolean =
false):void
{
super.loadBackground(battleModel,
```

```
battleModel);
//focus the view
//is there a location we want to focus on?
var foundFleet:Boolean;
if (_sectorModel.focusLocation.x != 0)
{
jumpToLocation(_sectorModel.focusLocation.x, _sectorModel.focusLocation.y);
} else
{
//focus on the fleet if there is one
if (_sectorModel.focusFleetID && !_sectorModel.viewBase)
{
if (selectFleet(_sectorModel.focusFleetID, true, false, false))
foundFleet = true;
}
_sectorModel.viewBase = false;
if (!foundFleet)
{
//finally focus on the starbase if the fleet doesn't exist
var base:BaseVO = _starbaseModel.currentBase;
if (base)
jumpToLocation(base.sectorLocationX, base.sectorLocationY);
}
}
}

public function getTransgateDestinations():Vector.<SectorVO>
{
return _sectorModel.destinations;
}

public function getPrivateDestinations():Vector.<SectorVO>
{
return _sectorModel.privateDestinations;
}

public function getConstantPrototypeByName( v:String ):*
{
return _prototypeModel.getConstantPrototypeValueByName(v);
}

public function getTransgateCustomDestinationPrototype( key:String ):IPrototype
{
return _prototypeModel.getTransgateCustomDestinationPrototype(key);
}

public function getTransgateCustomDestinationGroupByCustomDestinationGroup( group:String
):Vector.<IPrototype>
{
return
_prototypeModel.getTransgateCustomDestinationGroupByCustomDestinationGroup(group);
```

```
}

public function getPlayer( id:String ):PlayerVO { return _playerModel.getPlayer(id); }

public function getBase( id:String ):BaseVO { return _starbaseModel.getBaseByID(id); }
public function getFleets():Vector.<FleetVO> { return _fleetModel.fleets; }

public function addInteractListener( listener:Function ):void { _onInteractSignal.add(listener); }
public function removeInteractListener( listener:Function ):void {
_onInteractSignal.remove(listener); }

public function addListenerOnCoordsUpdate( listener:Function ):void {
_system.onCoordsUpdate.add(listener); }
public function removeListenerOnCoordsUpdate( listener:Function ):void {
_system.onCoordsUpdate.remove(listener); }

public function addBattleListener( listener:Function ):void { _onBattleSignal.add(listener); }
public function removeBattleListener( listener:Function ):void { _onBattleSignal.remove(listener);
}

public function addListenerForFleetUpdate( listener:Function ):void {
_fleetModel.onUpdatedFleetsSignal.add(listener); }
public function removeListenerForFleetUpdate( listener:Function ):void {
_fleetModel.onUpdatedFleetsSignal.remove(listener); }

public function addNotificationListener( listener:Function ):void {
_onNotificationSignal.add(listener); }
public function removeNotificationListener( listener:Function ):void {
_onNotificationSignal.remove(listener); }

public function addSelectionChangeListener( listener:Function ):void {
_system.onSelectionChangeSignal.add(listener); }

public function addOnGenericAllianceMessageRecievedListener( callback:Function ):void {
_allianceModel.onGenericAllianceMessageRecieved.add(callback); }
public function removeOnGenericAllianceMessageRecievedListener( callback:Function ):void {
_allianceModel.onGenericAllianceMessageRecieved.remove(callback); }

public function jumpToLocation( x:Number, y:Number ):void
{
_system.jumpToLocation(x, y);
}

public function get currentMission():MissionVO { return _missionModel.currentMission; }
public function get neighborhood():int { return _sectorModel.neighborhood; }
public function get focusFleetID():String { return _sectorModel.focusFleetID; }
public function get selectedEntity():Entity { return _system.selected; }
public function get selectedEnemy():Entity { return _system.selectedEnemy; }
public function get sectorID():String { return _sectorModel.sectorID; }
public
```

```
function get sectorName():String { return _sectorModel.sectorName; }
public function get sectorEnum():String { return _sectorModel.sectorEnum; }

[Inject]
public function set missionModel( v:MissionModel ):void { _missionModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set allianceModel( v:AllianceModel ):void { _allianceModel = v; }

override public function destroy():void
{
super.destroy();
_destinations = null;
_onBattleSignal.removeAll();
_onBattleSignal = null;
_onInteractSignal.removeAll();
_onInteractSignal = null;
_onNotificationSignal.removeAll();
_onNotificationSignal = null;
_fleetModel = null;
_missionModel = null;
_sectorModel = null;
_serverController = null;
_starbaseModel = null;
_system.presenter = null;
_system = null;
}
}
}
```

File 367: igw\com\presenter\shared\AchievementPresenter.as

```
package com.presenter.shared
{
import com.controller.GameController;
import com.model.achievements.AchievementModel;
import
```

```
com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BaseVO;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;

public class AchievementPresenter extends ImperiumPresenter implements
IAchievementPresenter
{
private var _assetModel:AssetModel;
private var _prototypeModel:PrototypeModel;
private var _achievementModel:AchievementModel;
private var _starbaseModel:StarbaseModel;
private var _gameController:GameController;

public function getAchievementPrototypes():Vector.<IPrototype>
{
return _prototypeModel.getAchievementPrototypes();
}

public function getFilterAchievementPrototypes():Vector.<IPrototype>
{
return _prototypeModel.getFilterAchievementPrototypes();
}

public function getScoreValueByName( v:String ):uint
{
return _achievementModel.getScoreValueByName(v);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function getAssetVOFromIPrototype( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function requestAchievements():void
{
_gameController.requestAchievements();
}

public
```

```
function requestAllScores():void
{
_gameController.requestAllScores();
}

public function claimAchievementReward( achievement:String ):void
{
_gameController.claimAchievementReward(achievement);
}

public function maxCredits():int
{
var currentBase:BaseVO = _starbaseModel.currentBase;
if (currentBase)
return currentBase.maxCredits;

return 0;
}

public function onAddAchievementsUpdatedListener( Listener:Function ):void {
_achievementModel.onAchievementsUpdated.add(Listener); }
public function onRemoveAchievementsUpdatedListener( Listener:Function ):void {
_achievementModel.onAchievementsUpdated.remove(Listener); }

public function onAddAllScoresUpdatedListener( Listener:Function ):void {
_achievementModel.onAllScoresUpdated.add(Listener); }
public function onRemoveAllScoresUpdatedListener( Listener:Function ):void {
_achievementModel.onAllScoresUpdated.remove(Listener); }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set achievementModel( v:AchievementModel ):void { _achievementModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }
}
}

-----------------------------------
File 368: igw\com\presenter\shared\AlliancePresenter.as
package com.presenter.shared
{
import com.controller.ServerController;
import com.controller.GameController;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import
```

```
com.model.alliance.AllianceModel;
import com.model.player.CurrentUser;
import com.presenter.ImperiumPresenter;
import com.service.server.outgoing.alliance.AllianceCreateAllianceRequest;
import com.service.server.outgoing.alliance.AllianceDemoteRequest;
import com.service.server.outgoing.alliance.AllianceIgnoreInvitesRequest;
import com.service.server.outgoing.alliance.AllianceJoinRequest;
import com.service.server.outgoing.alliance.AllianceKickRequest;
import com.service.server.outgoing.alliance.AllianceLeaveRequest;
import com.service.server.outgoing.alliance.AlliancePromoteRequest;
import com.service.server.outgoing.alliance.AllianceRequestBaselineRequest;
import com.service.server.outgoing.alliance.AllianceRequestPublicsRequest;
import com.service.server.outgoing.alliance.AllianceRequestRosterRequest;
import com.service.server.outgoing.alliance.AllianceSendInviteRequest;
import com.service.server.outgoing.alliance.AllianceSetDescriptionRequest;
import com.service.server.outgoing.alliance.AllianceSetMOTDRequest;
import com.service.server.outgoing.alliance.AllianceSetPublicRequest;

import flash.utils.Dictionary;

public class AlliancePresenter extends ImperiumPresenter implements IAlliancePresenter
{
private var _allianceModel:AllianceModel;
private var _serverController:ServerController;
private var _gameController:GameController;

[PostConstruct]
override public function init():void
{
super.init();
}

public function allianceBaselineRequest( allianceKey:String ):void
{
var getAllianceBaseline:AllianceRequestBaselineRequest =
AllianceRequestBaselineRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_REQUEST_BASELINE));
getAllianceBaseline.allianceKey = allianceKey;
_serverController.send(getAllianceBaseline);
}

public function allianceRosterRequest( allianceKey:String ):void
{
var getAllianceRoster:AllianceRequestRosterRequest =
AllianceRequestRosterRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_REQUEST_ROSTER));
getAllianceRoster.allianceKey = allianceKey;
_serverController.send(getAllianceRoster);
}

public
```

```
function alliancePublicAllianceRequest():void
{
var getPublics:AllianceRequestPublicsRequest =
AllianceRequestPublicsRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_REQUEST_PUBLICS));
getPublics.faction = CurrentUser.faction;
_serverController.send(getPublics);
}

public function allianceCreateRequest( name:String, isPublic:Boolean, description:String ):void
{
var createAlliance:AllianceCreateAllianceRequest =
AllianceCreateAllianceRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_CREATE_ALLIANCE));
createAlliance.name = name;
createAlliance.publicAlliance = isPublic;
createAlliance.description = description;
_serverController.send(createAlliance);
}

public function allianceSetMOTD( motd:String ):void
{
var setMOTD:AllianceSetMOTDRequest =
AllianceSetMOTDRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SET_MOTD));
setMOTD.motd = motd;
_serverController.send(setMOTD);
}

public function allianceSetDescription( description:String ):void
{
var setDescription:AllianceSetDescriptionRequest =
AllianceSetDescriptionRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SET_DESCRIPTION));
setDescription.description = description;
_serverController.send(setDescription);
}

public function allianceSetPublic( isPublic:Boolean ):void
{
var setPublic:AllianceSetPublicRequest =
AllianceSetPublicRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SET_PUBLIC));
setPublic.publicAlliance = isPublic;
_serverController.send(setPublic);
}

public function alliancePlayerPromote( playerKey:String ):void
{
var promotePlayer:AlliancePromoteRequest =
AlliancePromoteRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
```

```
RequestEnum.ALLIANCE_PROMOTE));
promotePlayer.playerKey = playerKey;
_serverController.send(promotePlayer);
}

public function alliancePlayerDemote( playerKey:String ):void
{
var demotePlayer:AllianceDemoteRequest =
AllianceDemoteRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_DEMOTE));
demotePlayer.playerKey = playerKey;
_serverController.send(demotePlayer);
}

public function alliancePlayerKick( playerKey:String ):void
{
var kickPlayer:AllianceKickRequest =
AllianceKickRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_KICK));
kickPlayer.playerKey = playerKey;
_serverController.send(kickPlayer);
}

public function allianceLeave():void
{
var leaveAlliance:AllianceLeaveRequest =
AllianceLeaveRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_LEAVE_ALLIANCE));
_serverController.send(leaveAlliance);
}

public function allianceJoin( allianceKey:String ):void
{
var join:AllianceJoinRequest =
AllianceJoinRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_JOIN_ALLIANCE));
join.allianceKey = allianceKey;
_serverController.send(join);
}

public function allianceSendInvite( playerKey:String ):void
{
var sendInvite:AllianceSendInviteRequest =
AllianceSendInviteRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SEND_INVITE));
sendInvite.playerKey = playerKey;
_serverController.send(sendInvite);
}

public function allianceIgnoreInvites():void
{
```

```
var ignoreInvites:AllianceIgnoreInvitesRequest =
AllianceIgnoreInvitesRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_IGNORE_INVITES));
_serverController.send(ignoreInvites);
}

public function getAllianceInvites():Dictionary
{
return _allianceModel.getAllianceInvites();
}

public function sendGetMailboxMessage():void
{
_gameController.mailGetMailbox();
}

public function addOnAllianceUpdatedListener( callback:Function ):void {
_allianceModel.onAllianceUpdated.add(callback); }
public function removeOnAllianceUpdatedListener( callback:Function ):void {
_allianceModel.onAllianceUpdated.remove(callback); }

public function addOnOpenAlliancesUpdatedListener( callback:Function ):void {
_allianceModel.onOpenAlliancesUpdated.add(callback); }
public function removeOnOpenAlliancesUpdatedListener( callback:Function ):void {
_allianceModel.onOpenAlliancesUpdated.remove(callback); }

public function addOnInvitedAlliancesUpdatedListener( callback:Function ):void {
_allianceModel.onInvitedAlliancesUpdated.add(callback); }
public function removeOnInvitedAlliancesUpdatedListener( callback:Function ):void {
_allianceModel.onInvitedAlliancesUpdated.remove(callback); }

public function addOnAllianceMembersUpdatedListener( callback:Function ):void {
_allianceModel.onAllianceMembersUpdated.add(callback); }
public function removeOnAllianceMembersUpdatedListener( callback:Function ):void {
_allianceModel.onAllianceMembersUpdated.remove(callback); }

public function addOnGenericAllianceMessageRecievedListener( callback:Function ):void {
_allianceModel.onGenericAllianceMessageRecieved.add(callback); }
public function removeOnGenericAllianceMessageRecievedListener( callback:Function ):void {
_allianceModel.onGenericAllianceMessageRecieved.remove(callback); }

[Inject]
public function set allianceModel( v:AllianceModel ):void { _allianceModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }

override public function destroy():void
{
```

```
super.destroy();
_allianceModel = null;
_serverController = null;
}
}
}

------------------------------------
File 369: igw\com\presenter\shared\BookmarkPresenter.as
package com.presenter.shared
{
import com.controller.GameController;
import com.controller.ServerController;
import com.enum.server.OrderEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.SectorEvent;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.player.BookmarkVO;
import com.model.player.CurrentUser;
import com.model.sector.SectorModel;
import com.presenter.ImperiumPresenter;
import com.service.server.outgoing.sector.SectorOrderRequest;

import org.ash.core.Game;

public class BookmarkPresenter extends ImperiumPresenter implements IBookmarkPresenter
{
private var _game:Game;
private var _sectorModel:SectorModel;
private var _fleetModel:FleetModel;
private var _gameController:GameController;
private var _serverController:ServerController;
private var _system:SectorInteractSystem;

public function deleteBookmark( index:uint ):void
{
CurrentUser.removeBookmark(index);
_gameController.bookmarkDelete(index);
}

public function updateBookmark( bookmark:BookmarkVO ):void
{
_gameController.bookmarkUpdate(bookmark);
}

public function fleetGotoCoords( x:int, y:int, sector:String ):void
{
```

```
if (_sectorModel.focusFleetID != null && _sectorModel.focusFleetID != '')
{
var order:SectorOrderRequest =
SectorOrderRequest(_serverController.getRequest(ProtocolEnum.SECTOR_CLIENT,
RequestEnum.SECTOR_ISSUE_ORDER));
order.entityId = _sectorModel.focusFleetID;
order.orderType = OrderEnum.REMOTE_MOVE;
order.destinationSector = sector;
order.targetLocationX = x;
order.targetLocationY = y;
_serverController.send(order);
}
}

public function hasSelectedFleet():Boolean
{
var hasASelectedFleet:Boolean;

if (_sectorModel.focusFleetID != null && _sectorModel.focusFleetID != '')
{
var fleet:FleetVO = _fleetModel.getFleet(_sectorModel.focusFleetID);
if (fleet && fleet.sector != '')
hasASelectedFleet = true;
}

return hasASelectedFleet;
}

public function gotoCoords( x:int, y:int, sector:String ):void
{

if (sector != _sectorModel.sectorID)
{
jumpToSector(x, y, sector);
} else
{
var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system != null)
system.moveToLocation(x, y);
else
{
jumpToSector(x, y, sector);
}
}

}

private function jumpToSector( x:int, y:int, sector:String ):void
{
```

```
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sector, null,
x, y);
dispatch(sectorEvent);
}

[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }

override public function destroy():void
{
super.destroy();
}
}
}
```

File 370: igw\com\presenter\shared\ChatPresenter.as

```
package com.presenter.shared
{
import com.Application;
import com.controller.ChatController;
import com.controller.GameController;
import com.event.SectorEvent;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.chat.ChatChannelVO;
import com.model.chat.ChatPanelVO;
import com.model.motd.MotDModel;
import com.model.motd.MotDVO;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.sector.SectorModel;
import com.presenter.ImperiumPresenter;

import com.service.ExternalInterfaceAPI;

import com.event.ServerEvent;

import flash.events.IEventDispatcher;

import flash.utils.Dictionary;

import
```

```
org.ash.core.Game;

public class ChatPresenter extends ImperiumPresenter implements IChatPresenter
{
private var _game:Game;
private var _chatController:ChatController;
private var _gameController:GameController;
private var _sectorModel:SectorModel;
private var _motdModel:MotDModel;
private var _playerModel:PlayerModel;

public function sendChatMessage( message:String ):void
{
if (Application.NETWORK == Application.NETWORK_GUEST)
{
ExternalInterfaceAPI.logConsole("Guest Chat Restriction");
var serverEvent:ServerEvent
serverEvent = new ServerEvent(ServerEvent.GUEST_RESTRICTION);
_eventDispatcher.dispatchEvent(serverEvent);
}
else
{
if (message.charAt(0) == '/')
_chatController.handleSlashCommand(message);
else
_chatController.sendChatMessageWithDefaultChannel(message);
}
}

public function gotoCoords( x:int, y:int, sector:String ):void
{

if (sector != _sectorModel.sectorID)
{
jumpToSector(x, y, sector);
} else
{
var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system != null)
system.moveToLocation(x, y);
else
{
jumpToSector(x, y, sector);
}
}

}

private function jumpToSector( x:int, y:int, sector:String ):void
{
```

```
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sector, null,
x, y);
dispatch(sectorEvent);
}

public function requestPlayer( id:String, name:String = '' ):void
{
_gameController.leaderboardRequestPlayerProfile(id, name);
}

public function blockOrUnblockPlayer( id:String ):void
{
_chatController.sendUnblockOrBlock(id);
}

public function isBlocked( id:String ):Boolean
{
return _chatController.isBlocked(id);
}

public function mutePlayer( id:String ):void
{
_chatController.mutePlayer(id);
}

public function isMuted( id:String ):Boolean
{
return _chatController.isMuted(id);
}

public function getChannelColorFromChannelID( channelID:int ):uint
{
return _chatController.getChannelColorFromId(channelID);
}

public function getPlayer( id:String ):PlayerVO
{
return _playerModel.getPlayer(id);
}

public function getPanelLogs( panelID:uint ):String { return
_chatController.getPanelLogs(panelID); }
public function getActiveChannels():Dictionary { return _chatController.getActiveChannels(); }

public function addChatListener( callback:Function ):void {
_chatController.addChatListener(callback); }
public function removeChatListener( callback:Function ):void {
_chatController.removeChatListener(callback); }

public
```

```
function addOnActiveChannelUpdatedListener( callback:Function ):void {
_chatController.addOnActiveChannelUpdatedListener(callback); }
public function removeOnActiveChannelUpdatedListener( callback:Function ):void {
_chatController.removeOnActiveChannelUpdatedListener(callback); }

public function addOnDefaultChannelLoadedListener( callback:Function ):void {
_chatController.addOnDefaultChannelLoadedListener(callback); }
public function removeOnDefaultChannelLoadedListener( callback:Function ):void {
_chatController.removeOnDefaultChannelLoadedListener(callback); }

public function addOnDefaultChannelUpdatedListener( callback:Function ):void {
_chatController.addOnDefaultChannelUpdatedListener(callback); }
public function removeOnDefaultChannelUpdatedListener( callback:Function ):void {
_chatController.removeOnDefaultChannelUpdatedListener(callback); }

public function addMotDUpdatedListener( callback:Function ):void {
_motdModel.newMessage.add(callback); }
public function removeMotDUpdatedListener( callback:Function ):void {
_motdModel.newMessage.remove(callback); }

public function addOnPlayerVOAddedListener( callback:Function ):void {
_playerModel.onPlayerAdded.add(callback); }
public function removeOnPlayerVOAddedListener( callback:Function ):void {
_playerModel.onPlayerAdded.remove(callback); }

public function get defaultChannel():ChatChannelVO { return
_chatController.getDefaultChannel()}
public function get chatHasFocus():Boolean { return _chatController.chatHasFocus; }
public function get blockedUsers():Vector.<String> { return _chatController.blockedList; }
public function get chatPanels():Vector.<ChatPanelVO> { return
_chatController.getChatPanels(); }
public function get motdMessages():Vector.<MotDVO> { return _motdModel.motd; }

public function set defaultChannel( newDefault:ChatChannelVO ):void {
_chatController.setDefaultChannel(newDefault); }
public function set chatHasFocus( value:Boolean ):void { _chatController.chatHasFocus = value;
}

[Inject]
public function set chatController( v:ChatController ):void { _chatController = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }
[Inject]
public function set motdModel( v:MotDModel ):void { _motdModel = v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
}
```

```
}

------------------------------------
File 371: igw\com\presenter\shared\CommandPresenter.as
package com.presenter.shared
{
import com.Application;
import com.event.BattleEvent;
import com.event.SectorEvent;
import com.event.StateEvent;
import com.game.entity.components.shared.Position;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.sector.SectorModel;
import com.model.starbase.BaseVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;

import org.ash.core.Entity;
import org.ash.core.Game;

public class CommandPresenter extends ImperiumPresenter implements ICommandPresenter
{
private var _assetModel:AssetModel;
private var _fleetModel:FleetModel;
private var _game:Game;
private var _sectorModel:SectorModel;
private var _starbaseModel:StarbaseModel;
private var _system:SectorInteractSystem;

[PostConstruct]
override public function init():void
{
super.init();

_system = SectorInteractSystem(_game.getSystem(SectorInteractSystem));
}

public function selectFleet( fleetID:String, gotoLocation:Boolean = true, canEnterBattle:Boolean
= true ):Boolean
{
var fleetVO:FleetVO = getFleetVO(fleetID);
var sectorEvent:SectorEvent;
if (fleetVO)
{
_sectorModel.focusFleetID
```

```
= fleetVO.id;
if (!_fteController.running && fleetVO.inBattle && canEnterBattle)
{
var battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN,
fleetVO.battleServerAddress);
dispatch(battleEvent);
} else if (fleetVO.sector != '')
{
var entity:Entity = _game.getEntity(fleetID);
if (_system != null && entity != null && fleetVO.sector == _sectorModel.sectorID)
{
_system.selectEntity(entity, gotoLocation);
} else
{
if ((fleetVO.sector != _sectorModel.sectorID || Application.STATE ==
StateEvent.GAME_STARBASE) && gotoLocation)
{
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, fleetVO.sector, fleetID);
dispatch(sectorEvent);
}
}
return true;
}
}
return false;
}

public function getEntity( fleetID:String ):Entity
{
return _game.getEntity(fleetID);
}

public function jumpToLocation( x:Number, y:Number ):void
{
if (_system != null)
_system.jumpToLocation(x, y);
}

public function getFleetVO( id:String ):FleetVO
{
return _fleetModel.getFleet(id);
}

public function loadIconImageFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
if(!_currentAssetVO)
return;
var icon:String = _currentAssetVO.iconImage;
loadIcon(icon, callback);
}
```

```
public function loadSmallImageFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
if(!_currentAssetVO)
return;
var icon:String = _currentAssetVO.smallImage;
loadIcon(icon, callback);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

override protected function onStateChange( e:StateEvent ):void
{
switch (e.type)
{
case StateEvent.GAME_SECTOR:
_system = SectorInteractSystem(_game.getSystem(SectorInteractSystem));
break;
case StateEvent.GAME_SECTOR_CLEANUP:
_system = null;
break;
}
super.onStateChange(e);
}

public function getBuildingVOByClass( itemClass:String, highestLevel:Boolean = false
):BuildingVO { return _starbaseModel.getBuildingByClass(itemClass, highestLevel); }

public function addListenerForFleetUpdate( listener:Function ):void {
_fleetModel.onUpdatedFleetsSignal.add(listener); }
public function removeListenerForFleetUpdate( listener:Function ):void {
_fleetModel.onUpdatedFleetsSignal.remove(listener); }

public function addSelectionChangeListener( listener:Function ):void { if (_system != null)
_system.onSelectionChangeSignal.add(listener); }
public function removeSelectionChangeListener( listener:Function ):void { if (_system != null)
_system.onSelectionChangeSignal.remove(listener); }

public function get sectorID():String { return _sectorModel.sectorID; }
public function get focusFleetID():String { return _sectorModel.focusFleetID; }
public function get fleets():Vector.<FleetVO> { return _fleetModel.fleets; }
public function get currentBase():BaseVO { return _starbaseModel.currentBase; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
```

```
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }

override public function destroy():void
{
super.destroy();
_assetModel = null;
_fleetModel = null;
_game = null;
_sectorModel = null;
_starbaseModel = null;
_system = null;
}
}
}

-------------------------------------
File 372: igw\com\presenter\shared\EngineeringPresenter.as
package com.presenter.shared
{
import com.controller.transaction.TransactionController;
import com.event.TransactionEvent;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionVO;
import com.presenter.ImperiumPresenter;

import flash.utils.Dictionary;

public class EngineeringPresenter extends ImperiumPresenter implements
IEngineeringPresenter
{
private var _assetModel:AssetModel;
private var _fleetModel:FleetModel;
private var _prototypeModel:PrototypeModel;
private var _starbaseModel:StarbaseModel;
private
```

```
var _transactionController:TransactionController;

[PostConstruct]
override public function init():void
{
super.init();
}

public function getAssetVO( name:String ):AssetVO { return _assetModel.getEntityData(name); }
public function getBaseRepairTransaction():TransactionVO { return
_transactionController.getBaseRepairTransaction(); }
public function getBuildingCount( buildingClass:String ):int { return
_starbaseModel.currentBase.getBuildingCount(buildingClass); }
public function getBuildingByID( id:String ):BuildingVO { return
_starbaseModel.getBuildingByID(id, false); }
public function getBuildingPrototypeByClassAndLevel( itemClass:String, level:int ):IPrototype {
return _prototypeModel.getBuildingPrototypeByClassAndLevel(itemClass, level); }
public function getResearchByID( id:String ):ResearchVO { return
_starbaseModel.getResearchByID(id, false); }
public function getStarbaseBuildingTransaction( constructionCategory:String ):TransactionVO {
return _transactionController.getStarbaseBuildingTransaction(constructionCategory, null); }
public function getStarbaseResearchTransaction( buildingType:String ):TransactionVO { return
_transactionController.getStarbaseResearchTransactionByBuildingType(buildingType); }
public function loadIcon( url:String, callback:Function ):void {
_assetModel.getFromCache("assets/" + url, callback); }
public function getShipVOByID( id:String ):ShipVO { return _fleetModel.getShip(id); }
public function getRepairFleetByID( id:String ):FleetVO { return _fleetModel.getFleet(id); }

public function loadTransactionIcon( transaction:TransactionVO, callback:Function ):void
{
var assetVO:AssetVO;
switch (transaction.type)
{
case TransactionEvent.STARBASE_BUILD_SHIP:
case TransactionEvent.STARBASE_REFIT_SHIP:
var ship:ShipVO = _fleetModel.getShip(transaction.id);
if (ship)
{
assetVO = _assetModel.getEntityData(ship.asset);
loadIcon(assetVO.largeImage, callback);
}
break;

case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
case TransactionEvent.STARBASE_REFIT_BUILDING:
case TransactionEvent.STARBASE_REPAIR_BASE:
var building:BuildingVO = _starbaseModel.getBuildingByID(transaction.id, false);
if (building)
{
assetVO
```

```
= _assetModel.getEntityData(building.asset);
loadIcon(assetVO.mediumImage, callback);
}
break;

case TransactionEvent.STARBASE_REPAIR_FLEET:
var fleet:FleetVO = _fleetModel.getFleet(transaction.id);
if (fleet)
{
ship = fleet.ships[0];
if (ship)
{
assetVO = _assetModel.getEntityData(ship.asset);
loadIcon(assetVO.largeImage, callback);
}
}
break;

case TransactionEvent.STARBASE_RESEARCH:
var research:ResearchVO = _starbaseModel.getResearchByID(transaction.id);
if (research)
{
assetVO = _assetModel.getEntityData(research.uiAsset);
loadIcon(assetVO.mediumImage, callback);
}
break;
}
}

public function addTransactionListener( type:int, callback:Function ):void {
_transactionController.addListener(type, callback); }
public function removeTransactionListener( callback:Function ):void {
_transactionController.removeListener(callback); }

public function get transactions():Dictionary { return _transactionController.transactions; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

override public function destroy():void
{
super.destroy();
```

```
_assetModel = null;
_fleetModel = null;
_prototypeModel = null;
_starbaseModel = null;
_transactionController = null;
}
}
}

-----------------------------------
File 373: igw\com\presenter\shared\EventPresenter.as
package com.presenter.shared
{
import com.controller.GameController;
import com.model.achievements.AchievementModel;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.event.EventModel;
import com.model.event.EventVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.presenter.ImperiumPresenter;

public class EventPresenter extends ImperiumPresenter implements IEventPresenter
{
private var _eventModel:EventModel;
private var _achievementModel:AchievementModel;
private var _assetModel:AssetModel;
private var _prototypeModel:PrototypeModel;
private var _gameController:GameController;

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function getResearchItemPrototypeByName( v:String ):IPrototype
{
var researchProto:IPrototype = _prototypeModel.getResearchPrototypeByName(v);
var refName:String;

if (researchProto)
refName = researchProto.getValue("referenceName");
else
refName = v;

var iproto:IPrototype = _prototypeModel.getShipPrototype(refName);
if (!iproto)
iproto
```

```
= _prototypeModel.getWeaponPrototype(refName);
return iproto;

return null;
}

public function getAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function getScoreValueByName( v:String ):uint
{
return _achievementModel.getScoreValueByName(v);
}

public function requestAchievements():void
{
_gameController.requestAchievements();
}

public function requestAllScores():void
{
_gameController.requestAllScores();
}

public function addEventUpdatedListener( callback:Function ):void {
_eventModel.onEventsUpdated.add(callback); }
public function removeEventUpdatedListener( callback:Function ):void {
_eventModel.onEventsUpdated.remove(callback); }

public function onAddAchievementsUpdatedListener( Listener:Function ):void {
_achievementModel.onAchievementsUpdated.add(Listener); }
public function onRemoveAchievementsUpdatedListener( Listener:Function ):void {
_achievementModel.onAchievementsUpdated.remove(Listener); }

public function onAddAllScoresUpdatedListener( Listener:Function ):void {
_achievementModel.onAllScoresUpdated.add(Listener); }
public function onRemoveAllScoresUpdatedListener( Listener:Function ):void {
_achievementModel.onAllScoresUpdated.remove(Listener); }

public function get currentActiveEvent():EventVO { return _eventModel.currentActiveEvent; }
public function get activeEvents():Vector.<EventVO> { return _eventModel.activeEvents; }
public function get upcomingEvents():Vector.<EventVO> { return _eventModel.upcomingEvents;
}

[Inject]
public
```

```
function set eventModel( v:EventModel ):void { _eventModel = v; }

[Inject]
public function set achievementModel( v:AchievementModel ):void { _achievementModel = v; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }

}
}

-------------------------------------
File 374: igw\com\presenter\shared\GameOfChancePresenter.as
package com.presenter.shared
{
import com.controller.transaction.TransactionController;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.battle.BattleModel;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintModel;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.presenter.ImperiumPresenter;

public class GameOfChancePresenter extends ImperiumPresenter implements
IGameOfChancePresenter
{

private var _transactionController:TransactionController;

private var _assetModel:AssetModel;
private var _blueprintModel:BlueprintModel;
private var _battleModel:BattleModel;
private var _prototypeModel:PrototypeModel;

[PostConstruct]
override public function init():void
{
super.init();
}

public function loadIcon( url:String, callback:Function ):void
{
```

```
_assetModel.getFromCache("assets/" + url, callback);
}

public function getAssetVO( assetName:String ):AssetVO
{
return _assetModel.getEntityData(assetName);
}

public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getConstantPrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getConstantPrototypeByName(name);
}

public function getBlueprintByName( prototype:String ):BlueprintVO
{
return _blueprintModel.getBlueprintByName(prototype);
}

public function getBlueprintPrototypeByName( v:String ):IPrototype
{
return _prototypeModel.getBlueprintPrototype(v);
}

public function getResearchPrototypeByName( v:String ):IPrototype
{
return _prototypeModel.getResearchPrototypeByName(v);
}

public function getShipPrototype( v:String ):IPrototype
{
return _prototypeModel.getShipPrototype(v);
}

public function getModulePrototypeByName( v:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getShipPrototype(v);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(v);
return iproto;
}

public function getBlueprintByID( id:String ):BlueprintVO
{
return
```

```
_blueprintModel.getBlueprintByID(id);
}

public function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number
{
return _transactionController.getBlueprintHardCurrencyCost(blueprint, partsPurchased);
}

public function removeBlueprintByName( name:String ):void
{
_blueprintModel.removeBlueprintByName(name);
}

public function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void
{
_transactionController.buyBlueprintTransaction(blueprint, partsPurchased);
}
public function completeBlueprintResearch( blueprint:BlueprintVO):void
{
_transactionController.completeBlueprintResearchTransaction(blueprint);
}

public function purchaseReroll( battleKey:String ):void
{
_transactionController.starbasePurchaseReroll(battleKey);
}

public function purchaseDeepScan( battleKey:String ):void
{
_transactionController.starbasePurchaseDeepScan(battleKey);
}

public function removeRerollFromAvailable( battleID:String ):void
{
_battleModel.removeRerollByID(battleID);
}

public function getAvailableRerolls():Vector.<BattleRerollVO>
{
return _battleModel.getAllAvailableRerolls();
}

public function addAvailableRerollUpdatedListener( callback:Function ):void {
_battleModel.onRerollAdded.add(callback); }
public function removeAvailableRerollUpdatedListener( callback:Function ):void {
_battleModel.onRerollAdded.remove(callback); }

public function addRerollFromRerollCallback( callback:Function ):void {
_battleModel.onRerollUpdated.add(callback); }
public
```

```
function removeRerollFromRerollCallback( callback:Function ):void {
_battleModel.onRerollUpdated.remove(callback); }

public function addRerollFromScanCallback( callback:Function ):void {
_battleModel.onRerollUpdated.add(callback); }
public function removeRerollFromScanCallback( callback:Function ):void {
_battleModel.onRerollUpdated.remove(callback); }

[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set blueprintModel( v:BlueprintModel ):void { _blueprintModel = v; }
[Inject]
public function set battleModel( value:BattleModel ):void { _battleModel = value; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
}
}

-----------------------------------
File 375: igw\com\presenter\shared\GamePresenter.as
package com.presenter.shared
{
import com.controller.GameController;
import com.event.TransitionEvent;
import com.game.entity.systems.shared.background.BackgroundSystem;
import com.game.entity.systems.shared.grid.GridSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.scene.SceneModel;
import com.presenter.ImperiumPresenter;

import com.model.battle.BattleModel;

import org.ash.core.Entity;
import org.ash.core.Game;
import org.osflash.signals.Signal;

public class GamePresenter extends ImperiumPresenter implements IGamePresenter
{
protected var _assetModel:AssetModel;
protected var _cleanupSignal:Signal;
protected var _game:Game;
protected var _gameController:GameController;
protected
```

```
var _prototypeModel:PrototypeModel;
protected var _sceneModel:SceneModel;

override public function init():void
{
super.init();
removeContextListeners();
_cleanupSignal = new Signal();
}

public function confirmReady():void
{
dispatch(new TransitionEvent(TransitionEvent.TRANSITION_COMPLETE));
_fteController.ready = true;
if (_fteController.progressStepOnStateChange)
_fteController.nextStep();
}

public function cleanup():void
{
//send out a cleanup signal to notify view to destroy themselves.
//once all views are finished being destroyed the presenter will automatically be destroyed as
well
if (_cleanupSignal)
_cleanupSignal.dispatch();
}

public function addCleanupListener( callback:Function ):void {
_cleanupSignal.addOnce(callback); }
public function removeCleanupListener( callback:Function ):void {
_cleanupSignal.remove(callback); }

public function loadBackground(battleModel:BattleModel, useModelData:Boolean = false):void
{
var bgSystem:BackgroundSystem =
BackgroundSystem(_game.getSystem(BackgroundSystem));
bgSystem.addReadySignal(confirmReady);
bgSystem.buildBackground(battleModel, useModelData);
var gridSystem:GridSystem = GridSystem(_game.getSystem(GridSystem));
gridSystem.onBackgroundReady();
}

public function getEntity( id:String ):Entity { return _game.getEntity(id); }

public function getAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}
```

```
public function getAssetVOByName( name:String ):AssetVO
{
return _assetModel.getEntityData(name);
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set sceneModel( v:SceneModel ):void { _sceneModel = v; }

override public function destroy():void
{
super.destroy();
_cleanupSignal.removeAll();
_cleanupSignal = null;

_assetModel = null;
_sceneModel = null;
_game = null;
_gameController = null;
_prototypeModel = null;
}
}
}

------------------------------------
File 376: igw\com\presenter\shared\IAchievementPresenter.as
package com.presenter.shared
{
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;

public interface IAchievementPresenter extends IImperiumPresenter
{
function getAchievementPrototypes():Vector.<IPrototype>;
function getFilterAchievementPrototypes():Vector.<IPrototype>;
function loadIcon( url:String, callback:Function ):void;
function getAssetVOFromIPrototype( prototype:IPrototype ):AssetVO;
function requestAchievements():void;
function requestAllScores():void;
function claimAchievementReward( achievement:String ):void;
function
```

```
maxCredits():int;
function getScoreValueByName( v:String ):uint
function onAddAchievementsUpdatedListener( Listener:Function ):void;
function onRemoveAchievementsUpdatedListener( Listener:Function ):void;
function onAddAllScoresUpdatedListener( Listener:Function ):void;
function onRemoveAllScoresUpdatedListener( Listener:Function ):void;

}
}

-----------------------------------
File 377: igw\com\presenter\shared\IAlliancePresenter.as
package com.presenter.shared
{
import com.presenter.IImperiumPresenter;

import flash.utils.Dictionary;

public interface IAlliancePresenter extends IImperiumPresenter
{
function allianceBaselineRequest( allianceKey:String ):void;
function allianceRosterRequest( allianceKey:String ):void;
function alliancePublicAllianceRequest():void;
function allianceCreateRequest( name:String, isPublic:Boolean, description:String ):void;
function allianceSetMOTD( motd:String ):void;
function allianceSetDescription( description:String ):void;
function allianceSetPublic( isPublic:Boolean ):void;
function alliancePlayerPromote( playerKey:String ):void;
function alliancePlayerDemote( playerKey:String ):void;
function alliancePlayerKick( playerKey:String ):void;
function allianceLeave():void;
function allianceJoin( allianceKey:String ):void;
function allianceSendInvite( playerKey:String ):void;
function allianceIgnoreInvites():void;
function getAllianceInvites():Dictionary;

function addOnAllianceUpdatedListener( callback:Function ):void;
function removeOnAllianceUpdatedListener( callback:Function ):void;
function addOnOpenAlliancesUpdatedListener( callback:Function ):void;
function removeOnOpenAlliancesUpdatedListener( callback:Function ):void;
function addOnInvitedAlliancesUpdatedListener( callback:Function ):void;
function removeOnInvitedAlliancesUpdatedListener( callback:Function ):void;
function addOnAllianceMembersUpdatedListener( callback:Function ):void;
function removeOnAllianceMembersUpdatedListener( callback:Function ):void;
function addOnGenericAllianceMessageRecievedListener( callback:Function ):void;
function removeOnGenericAllianceMessageRecievedListener( callback:Function ):void;

function sendGetMailboxMessage():void;
}
}

-----------------------------------
```

File 378: igw\com\presenter\shared\IBookmarkPresenter.as

```
package com.presenter.shared
{
import com.model.player.BookmarkVO;
import com.presenter.IImperiumPresenter;

public interface IBookmarkPresenter extends IImperiumPresenter
{
function gotoCoords( x:int, y:int, sector:String ):void;
function fleetGotoCoords( x:int, y:int, sector:String ):void
function deleteBookmark( index:uint ):void;
function updateBookmark( bookmark:BookmarkVO ):void;
function hasSelectedFleet():Boolean;
}
}
```

File 379: igw\com\presenter\shared\IChatPresenter.as

```
package com.presenter.shared
{
import com.model.chat.ChatChannelVO;
import com.model.chat.ChatPanelVO;
import com.model.motd.MotDVO;
import com.model.player.PlayerVO;
import com.presenter.IImperiumPresenter;

import flash.utils.Dictionary;

public interface IChatPresenter extends IImperiumPresenter
{
function sendChatMessage( message:String ):void;
function gotoCoords( x:int, y:int, sector:String ):void;

function getChannelColorFromChannelID( channelID:int ):uint;
function getActiveChannels():Dictionary;

function requestPlayer( id:String, name:String = '' ):void;

function blockOrUnblockPlayer( id:String ):void;
function isBlocked( id:String ):Boolean;
function mutePlayer( id:String ):void;
function isMuted( id:String ):Boolean;
function getPanelLogs( panelID:uint ):String;
function getPlayer( id:String ):PlayerVO;

function addChatListener( callback:Function ):void;
function removeChatListener( callback:Function ):void;
function addOnActiveChannelUpdatedListener( callback:Function ):void;
function removeOnActiveChannelUpdatedListener( callback:Function ):void;
function
```

```
addOnDefaultChannelLoadedListener( callback:Function ):void;
function removeOnDefaultChannelLoadedListener( callback:Function ):void;
function addOnDefaultChannelUpdatedListener( callback:Function ):void;
function removeOnDefaultChannelUpdatedListener( callback:Function ):void;
function addMotDUpdatedListener( callback:Function ):void;
function removeMotDUpdatedListener( callback:Function ):void;
function addOnPlayerVOAddedListener( callback:Function ):void;
function removeOnPlayerVOAddedListener( callback:Function ):void;

function get chatHasFocus():Boolean;
function get defaultChannel():ChatChannelVO;
function get blockedUsers():Vector.<String>;
function get chatPanels():Vector.<ChatPanelVO>;
function get motdMessages():Vector.<MotDVO>;

function set defaultChannel( newDefault:ChatChannelVO ):void;
function set chatHasFocus( value:Boolean ):void;
}
}

-----------------------------------
File 380: igw\com\presenter\shared\ICommandPresenter.as
package com.presenter.shared
{
import com.model.fleet.FleetVO;
import com.model.starbase.BaseVO;
import com.model.starbase.BuildingVO;
import com.presenter.IImperiumPresenter;

import org.ash.core.Entity;

public interface ICommandPresenter extends IImperiumPresenter
{
function selectFleet( fleetID:String, gotoLocation:Boolean = true, canEnterBattle:Boolean = true
):Boolean;
function jumpToLocation( x:Number, y:Number ):void;
function getEntity( fleetID:String ):Entity;
function getFleetVO( id:String ):FleetVO;
function loadIconImageFromEntityData( type:String, callback:Function ):void;
function loadSmallImageFromEntityData( type:String, callback:Function ):void;
function loadIcon( url:String, callback:Function ):void;

function getBuildingVOByClass( itemClass:String, highestLevel:Boolean = false ):BuildingVO;

function addListenerForFleetUpdate( listener:Function ):void;
function removeListenerForFleetUpdate( listener:Function ):void;

function addSelectionChangeListener( listener:Function ):void;
function removeSelectionChangeListener( listener:Function ):void;

function
```

```
get sectorID():String;
function get focusFleetID():String;
function get fleets():Vector.<FleetVO>;
function get currentBase():BaseVO;
}
}

------------------------------------
File 381: igw\com\presenter\shared\IEngineeringPresenter.as
package com.presenter.shared
{
import com.model.asset.AssetVO;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.transaction.TransactionVO;
import com.presenter.IImperiumPresenter;

import flash.utils.Dictionary;

public interface IEngineeringPresenter extends IImperiumPresenter
{
function getAssetVO( name:String ):AssetVO;
function getBaseRepairTransaction():TransactionVO;
function getBuildingCount( buildingClass:String ):int;
function getBuildingByID( id:String ):BuildingVO;
function getBuildingPrototypeByClassAndLevel( itemClass:String, level:int ):IPrototype;
function getResearchByID( id:String ):ResearchVO;
function getStarbaseBuildingTransaction( constructionCategory:String ):TransactionVO;
function getStarbaseResearchTransaction( buildingType:String ):TransactionVO;
function loadIcon( url:String, callback:Function ):void;
function getShipVOByID( id:String ):ShipVO;
function getRepairFleetByID( id:String ):FleetVO;
function loadTransactionIcon( transaction:TransactionVO, callback:Function ):void;

function addTransactionListener( type:int, callback:Function ):void;
function removeTransactionListener( callback:Function ):void;

function get transactions():Dictionary;
}
}

------------------------------------
File 382: igw\com\presenter\shared\IEventPresenter.as
package com.presenter.shared
{
import com.model.asset.AssetVO;
import com.model.event.EventVO;
import
```

```
com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;

public interface IEventPresenter extends IImperiumPresenter
{
function loadIcon( url:String, callback:Function ):void;
function getResearchItemPrototypeByName( v:String ):IPrototype;
function getAssetVO( prototype:IPrototype ):AssetVO;
function getScoreValueByName( v:String ):uint;
function requestAchievements():void;
function requestAllScores():void;

function addEventUpdatedListener( callback:Function ):void;
function removeEventUpdatedListener( callback:Function ):void;

function onAddAchievementsUpdatedListener( Listener:Function ):void;
function onRemoveAchievementsUpdatedListener( Listener:Function ):void;
function onAddAllScoresUpdatedListener( Listener:Function ):void;
function onRemoveAllScoresUpdatedListener( Listener:Function ):void;

function get currentActiveEvent():EventVO;
function get activeEvents():Vector.<EventVO>;
function get upcomingEvents():Vector.<EventVO>;
}
}

-----------------------------------
File 383: igw\com\presenter\shared\IGameOfChancePresenter.as
package com.presenter.shared
{
import com.model.asset.AssetVO;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;

public interface IGameOfChancePresenter extends IImperiumPresenter
{
function loadIcon( url:String, callback:Function ):void;
function getAssetVO( assetName:String ):AssetVO;
function getResearchPrototypeByName( v:String ):IPrototype;
function getConstantPrototypeValueByName( name:String ):Number;
function getConstantPrototypeByName( name:String ):IPrototype;
function getModulePrototypeByName( v:String ):IPrototype;
function getShipPrototype( v:String ):IPrototype;
function getBlueprintByName( prototype:String ):BlueprintVO;
function getBlueprintPrototypeByName( v:String ):IPrototype;
function getBlueprintByID( id:String ):BlueprintVO;
function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number;
function
```

```
removeBlueprintByName( name:String ):void;
function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void;
function purchaseReroll( battleKey:String ):void;
function purchaseDeepScan( battleKey:String ):void;
function removeRerollFromAvailable( battleID:String ):void;
function getAvailableRerolls():Vector.<BattleRerollVO>;

function addAvailableRerollUpdatedListener( callback:Function ):void;
function addRerollFromRerollCallback( callback:Function ):void;
function addRerollFromScanCallback( callback:Function ):void;

function removeAvailableRerollUpdatedListener( callback:Function ):void;
function removeRerollFromRerollCallback( callback:Function ):void;
function removeRerollFromScanCallback( callback:Function ):void;
}
}

------------------------------------
File 384: igw\com\presenter\shared\IGamePresenter.as
package com.presenter.shared
{
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;

import com.model.battle.BattleModel;

import org.ash.core.Entity;

public interface IGamePresenter extends IImperiumPresenter
{
function confirmReady():void;

function cleanup():void;

function loadBackground(battleModel:BattleModel, useModelData:Boolean = false):void;

function addCleanupListener( callback:Function ):void;
function removeCleanupListener( callback:Function ):void;

function getEntity( id:String ):Entity;
function getAssetVO( prototype:IPrototype ):AssetVO;
function getAssetVOByName( name:String ):AssetVO;
}
}

------------------------------------
File 385: igw\com\presenter\shared\ILeaderboardPresenter.as
package com.presenter.shared
{
import
```

```
com.model.asset.AssetVO;
import com.model.leaderboards.LeaderboardVO;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;

public interface ILeaderboardPresenter extends IImperiumPresenter
{
function getLeaderboardData():LeaderboardVO;

function getLeaderboardDataByType( type:int, scope:int ):LeaderboardVO;
function requestPlayer( id:String, name:String = '' ):void;
function addOnLeaderboardDataUpdatedListener( callback:Function ):void;
function removeOnLeaderboardDataUpdatedListener( callback:Function ):void;
function getFactionPrototypesByName( name:String ):IPrototype
function getRacePrototypeByName( name:String ):IPrototype;
function getCommendationRankPrototypesByName( name:String ):IPrototype;
function loadIcon( url:String, callback:Function ):void;
function getAssetVO( assetName:String ):AssetVO;
function getSectorName( sectorID:String ):String;
function getAssetVOFromIPrototype( prototype:IPrototype ):AssetVO;

function set currentLeaderboardType( v:int ):void;
function get currentLeaderboardType():int;

function set currentLeaderboardScope( v:int ):void;
function get currentLeaderboardScope():int;

}
}

-----------------------------------
File 386: igw\com\presenter\shared\IPlayerProfilePresenter.as
package com.presenter.shared
{
import com.model.asset.AssetVO;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;
import com.service.server.incoming.data.SectorData;

public interface IPlayerProfilePresenter extends IImperiumPresenter
{
function getPlayer( id:String ):PlayerVO;
function requestPlayer( id:String, name:String = '' ):void;
function loadPortraitProfile( portraitName:String, callback:Function ):void;
function loadSmallImage( portraitName:String, callback:Function ):void;
function reportPlayer( id:String ):void;
function allianceSendInvite( playerKey:String ):void;
function gotoCoords( x:int, y:int, sector:String ):void;
function getConstantPrototypeValueByName( name:String ):*;
function
```

```
renamePlayer( newName:String ):void;
function relocateStarbase( targetPlayer:String ):void;
function getAssetVO( name:String ):AssetVO;
function getCommendationRankPrototypesByName( name:String ):IPrototype;
function currentPlayerInABattle():Boolean;
function getCurrentUsersHomeSector():SectorData;
function hasSectorSplitTestCohort( sectorId:String ):Boolean;

function addOnPlayerVOAddedListener( callback:Function ):void;
function removeOnPlayerVOAddedListener( callback:Function ):void;
}
}

-----------------------------------
File 387: igw\com\presenter\shared\ITransitionPresenter.as
package com.presenter.shared
{
import com.event.StateEvent;
import com.presenter.IImperiumPresenter;

public interface ITransitionPresenter extends IImperiumPresenter
{
function sendEvents():void;

function addEvents( initEvent:StateEvent, cleanupEvent:StateEvent ):void;

function transitionComplete():void;

function addCompleteListener( callback:Function ):void;
function removeCompleteListener( callback:Function ):void;

function addUpdateListener( callback:Function ):void;
function removeUpdateListener( callback:Function ):void;
function updateView():void;

function get connectingText():String;

function get estimatedLoadCompleted():Number;
function get trackAnalytics():Boolean;

function set failed( v:Boolean ):void;
function get failed():Boolean;

function get hasWaiting():Boolean;
}
}

-----------------------------------
File 388: igw\com\presenter\shared\IUIPresenter.as
package com.presenter.shared
{
```

```
import com.controller.transaction.requirements.RequirementVO;
import com.model.asset.AssetVO;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintVO;
import com.model.event.EventVO;
import com.model.motd.MotDDailyRewardModel;
import com.model.motd.MotDModel;
import com.model.prototype.IPrototype;
import com.model.starbase.BuffVO;
import com.presenter.IImperiumPresenter;
import com.service.loading.LoadPriority;

import flash.utils.Dictionary;

import org.parade.core.IView;

public interface IUIPresenter extends IImperiumPresenter
{
function changeResolution():void;

function gotoCoords( x:int, y:int, sector:String ):void;
function viewBattleReplay( battleId:String ):void;

function getTransactions():Dictionary;
function loadIcon( url:String, callback:Function ):void;
function loadIconFromEntityData( type:String, callback:Function ):void;
function loadMessageImage( url:String, callback:Function ):void;
function loadPortraitSmall( portraitName:String, callback:Function ):void
function getAssetVOFromIPrototype( prototype:IPrototype ):AssetVO;
function getAssetVO( assetName:String ):AssetVO;
function getFilterAssetVO( prototype:IPrototype ):AssetVO;
function toggleSFXMute():void;
function toggleMusicMute():void;
function setSFXVolume( v:Number ):void;
function setMusicVolume( v:Number ):void;
function toggleFullScreen():void;
function fteNextStep():void;
function fteSkip():void;
function loadPortraitMedium( portraitName:String, callback:Function ):void;
function loadPortraitIcon( portraitName:String, callback:Function ):void;
function getPrototypeUIName( prototype:IPrototype ):String;
function getPrototypeUIIcon( prototype:IPrototype ):String;
function getFromCache( url:String, callback:Function = null, priority:int = LoadPriority.LOW,
absoluteURL:Boolean = false ):Object;

function addBattleLogListUpdatedListener( callback:Function ):void;
function removeBattleLogListUpdatedListener( callback:Function ):void;
function addBattleLogDetailUpdatedListener( callback:Function ):void;
function removeBattleLogDetailUpdatedListener( callback:Function ):void;

function
```

```
getBlueprintByName( prototype:String ):BlueprintVO;
function removeBlueprintByName( name:String ):void;
function getBlueprintByID( id:String ):BlueprintVO;
function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number;
function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void;
function purchaseReroll( battleKey:String ):void;
function purchaseDeepScan( battleKey:String ):void;
function addRerollFromRerollCallback( callback:Function ):void;
function addRerollFromScanCallback( callback:Function ):void;
function removeRerollFromAvailable( battleID:String ):void;
function getBattleEndDialogByFaction( faction:String, combatResult:String = 'Victory'
):Vector.<IPrototype>;
function getBlueprintPrototypeByName( name:String ):IPrototype;
function sendMailMessage( playerID:String, subject:String, body:String ):void;
function sendAllianceMailMessage( subject:String, body:String ):void;
function sendGetMailboxMessage():void;
function getFactionPrototypesByName( name:String ):IPrototype;
function getRacePrototypeByName( name:String ):IPrototype;
function requestPlayer( id:String, name:String = '' ):void;

function getBuffPrototypes():Dictionary;
function getBuffPrototypeByName( name:String ):IPrototype;
function getCommendationRankPrototypesByName( name:String ):IPrototype;
function getConstantPrototypeValueByName( name:String ):Number;
function linkCoords( x:int, y:int ):void;
function addBookmark( x:int, y:int ):void;
function addMailCountUpdateListener( callback:Function ):void;
function removeMailCountUpdateListener( callback:Function ):void;
function addOnMailHeadersUpdatedListener( callback:Function ):void;
function removeOnMailHeadersUpdatedListener( callback:Function ):void;
function addOnMailDetailUpdatedListener( callback:Function ):void;
function removeOnMailDetailUpdatedListener( callback:Function ):void;
function addMotDUpdatedListener( callback:Function ):void
function addDailyRewardListener( callback:Function ):void;
function removeDailyRewardListener( callback:Function ):void;
function removeMotDUpdatedListener( callback:Function ):void;
function addWarfrontUpdateListener( listener:Function ):void;
function removeWarfrontUpdateListener( listener:Function ):void;
function addAvailableRerollUpdatedListener( callback:Function ):void;
function removeAvailableRerollUpdatedListener( callback:Function ):void;
function removeRerollFromRerollCallback( callback:Function ):void;
function removeRerollFromScanCallback( callback:Function ):void;
function addEventUpdatedListener( callback:Function ):void;
function removeEventUpdatedListener( callback:Function ):void;

function sendMotDMessageRead( key:String ):void;
function allianceSendInvite( playerKey:String ):void;
function sendDailyClaimRequest( header:int, protocolID:int ):void;

function
```

```
getMailDetails( mailKey:String ):void;
function deleteMail( v:Vector.<String> ):void;
function mailRead( mailKey:String ):void;

function get unreadMailCount():uint;
function getPrototypeByName( proto:String ):IPrototype;
function getShipPrototypeByName( proto:String ):IPrototype;
function getFAQPrototypes():Vector.<IPrototype>;
function getBattleLogList( filter:String ):void;
function getBattleLogDetails( battleKey:String ):void;
function getAvailableRerolls():Vector.<BattleRerollVO>;

function addOnPlayerVOAddedListener( callback:Function ):void;
function removeOnPlayerVOAddedListener( callback:Function ):void;
function canEquip( prototype:IPrototype, slotType:String ):RequirementVO;

function addTransactionListener( type:int, callback:Function ):void;
function removeTransactionListener( callback:Function ):void;

function removeBuff( buff:BuffVO ):void;
function updateStarbasePlatform():void;
function getOfferPrototypeByName( name:String ):IPrototype;
function getOfferItemsByItemGroup( itemGroup:String ):Vector.<IPrototype>;
function getView( view:Class ):IView;

function get buffs():Vector.<BuffVO>;
function get bubbleTimeRemaining():Number;

function get isSFXMuted():Boolean;
function get isMusicMuted():Boolean;
function get sfxVolume():Number;
function get musicVolume():Number;
function get isFullScreen():Boolean;
function get motdModel():MotDModel;
function get motdDailyModel():MotDDailyRewardModel;
function get currentGameState():String;

function get igaContextMenuDefaultIndex():int;
function setIGAContextMenuDefaultIndex( v:int ):void;

function get tyrContextMenuDefaultIndex():int;
function setTYRContextMenuDefaultIndex( v:int ):void;

function get sovContextMenuDefaultIndex():int;
function setSOVContextMenuDefaultIndex( v:int ):void;

function get csContextMenuDefaultIndex():int;
function setCSContextMenuDefaultIndex( v:int ):void;

function get currentActiveEvent():EventVO;
function
```

```
get activeEvents():Vector.<EventVO>;
function get upcomingEvents():Vector.<EventVO>;
}
}

-----------------------------------
File 389: igw\com\presenter\shared\LeaderboardPresenter.as
package com.presenter.shared
{
import com.controller.GameController;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.leaderboards.LeaderboardModel;
import com.model.leaderboards.LeaderboardVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.presenter.ImperiumPresenter;

public class LeaderboardPresenter extends ImperiumPresenter implements
ILeaderboardPresenter
{
private var _gameController:GameController;
private var _leaderboardModel:LeaderboardModel;
private var _prototypeModel:PrototypeModel;
private var _sectorModel:SectorModel;
private var _assetModel:AssetModel;

public function getLeaderboardData():LeaderboardVO
{
var currentEntry:LeaderboardVO = _leaderboardModel.getLeaderboardData;

if (currentEntry == null || (currentEntry && currentEntry.needsUpdate()))
_gameController.leaderboardRequest(_leaderboardModel.lastRequestedType,
_leaderboardModel.lastLeaderboardScope);

return currentEntry;
}

public function getLeaderboardDataByType( type:int, scope:int ):LeaderboardVO
{
var currentEntry:LeaderboardVO = _leaderboardModel.getLeaderboardDataByType(type,
scope);

if (currentEntry == null || (currentEntry && currentEntry.needsUpdate()))
_gameController.leaderboardRequest(type, scope);

return currentEntry;
}

public
```

```
function requestPlayer( id:String, name:String = '' ):void
{
_gameController.leaderboardRequestPlayerProfile(id, name);
}

public function getFactionPrototypesByName( name:String ):IPrototype
{
return _prototypeModel.getFactionPrototypeByName(name);
}

public function getRacePrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getRacePrototypeByName(name);
}

public function getCommendationRankPrototypesByName( name:String ):IPrototype
{
return _prototypeModel.getCommendationRankPrototypesByName(name);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function getAssetVO( assetName:String ):AssetVO
{
return _assetModel.getEntityData(assetName);
}

public function getSectorName( sectorID:String ):String
{
return _sectorModel.getSectorNameFromID(sectorID);
}

public function getAssetVOFromIPrototype( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function addOnLeaderboardDataUpdatedListener( callback:Function ):void {
_leaderboardModel.onLeaderboardDataUpdated.add(callback); }
public function removeOnLeaderboardDataUpdatedListener( callback:Function ):void {
_leaderboardModel.onLeaderboardDataUpdated.remove(callback); }

public function set currentLeaderboardType( v:int ):void {
_leaderboardModel.lastRequestedType = v; }

public
```

```
function get currentLeaderboardType():int { return _leaderboardModel.lastRequestedType; }

public function set currentLeaderboardScope( v:int ):void {
_leaderboardModel.lastLeaderboardScope = v; }

public function get currentLeaderboardScope():int { return
_leaderboardModel.lastLeaderboardScope; }

[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }

[Inject]
public function set leaderboardModel( v:LeaderboardModel ):void { _leaderboardModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }

}
}

-----------------------------------
File 390: igw\com\presenter\shared\PlayerProfilePresenter.as
package com.presenter.shared
{
import com.controller.ChatController;
import com.controller.ServerController;
import com.controller.transaction.TransactionController;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.SectorEvent;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.model.sector.SectorVO;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;
import com.service.server.incoming.data.SectorData;
import com.service.server.outgoing.alliance.AllianceSendInviteRequest;
import com.service.server.outgoing.chat.ChatReportChatRequest;
import com.service.server.outgoing.leaderboard.LeaderboardRequestPlayerProfileRequest;

import org.ash.core.Game;

public
```

```
class PlayerProfilePresenter extends ImperiumPresenter implements IPlayerProfilePresenter
{
private var _playerModel:PlayerModel;
private var _assetModel:AssetModel;
private var _prototypeModel:PrototypeModel;
private var _sectorModel:SectorModel;
private var _starbaseModel:StarbaseModel;
private var _serverController:ServerController;
private var _chatController:ChatController;
private var _transactionController:TransactionController;
private var _game:Game;

[PostConstruct]
override public function init():void
{
super.init();
}

public function getPlayer( id:String ):PlayerVO
{
return _playerModel.getPlayer(id);
}

public function requestPlayer( id:String, name:String = '' ):void
{
var leaderboardRequestPlayerProfile:LeaderboardRequestPlayerProfileRequest =
LeaderboardRequestPlayerProfileRequest(_serverController.getRequest(ProtocolEnum.LEADERBOARD_
RequestEnum.
LEADERBOARD_REQUEST_PLAYER_PROFILE));
leaderboardRequestPlayerProfile.playerKey = id;
leaderboardRequestPlayerProfile.nameSearch = name;
_serverController.send(leaderboardRequestPlayerProfile);
}

public function allianceSendInvite( playerKey:String ):void
{
var sendInvite:AllianceSendInviteRequest =
AllianceSendInviteRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SEND_INVITE));
sendInvite.playerKey = playerKey;
_serverController.send(sendInvite);
}

public function reportPlayer( id:String ):void
{
var reportPlayerRequest:ChatReportChatRequest =
ChatReportChatRequest(_serverController.getRequest(ProtocolEnum.CHAT_CLIENT,
RequestEnum.CHAT_REPORT_CHAT));
reportPlayerRequest.playerKey = id;
_serverController.send(reportPlayerRequest);

if
```

```
(!_chatController.isMuted(id))
_chatController.mutePlayer(id);
}

public function gotoCoords( x:int, y:int, sector:String ):void
{

if (sector != _sectorModel.sectorID)
{
jumpToSector(x, y, sector);
} else
{
var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system != null)
system.moveToLocation(x, y);
else
{
jumpToSector(x, y, sector);
}
}

}

private function jumpToSector( x:int, y:int, sector:String ):void
{
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sector, null,
x, y);
dispatch(sectorEvent);
}

public function loadPortraitProfile( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(_assetModel.getFromCache(portraitName));
if(avatarVO != null)
_assetModel.getFromCache('assets/' + avatarVO.profileImage, callback);
}

public function loadSmallImage( smallImageName:String, callback:Function ):void
{
_assetModel.getFromCache('assets/' + smallImageName, callback);
}

public function getConstantPrototypeValueByName( name:String ):*
{
return _prototypeModel.getConstantPrototypeValueByName(name);
}

public function renamePlayer( newName:String ):void
{
_transactionController.starbaseRenamePlayer(newName);
```

```
}

public function relocateStarbase( targetPlayer:String ):void
{
_transactionController.starbaseRelocateStarbase(targetPlayer);
}

public function getAssetVO( assetName:String ):AssetVO
{
return _assetModel.getEntityData(assetName);
}

public function getCommendationRankPrototypesByName( name:String ):IPrototype
{
return _prototypeModel.getCommendationRankPrototypesByName(name);
}

public function currentPlayerInABattle():Boolean
{
var inBattle:Boolean;

if (_starbaseModel.currentBase.battleServerAddress != null &&
_starbaseModel.currentBase.battleServerAddress != '')
inBattle = true;

if (_starbaseModel.currentBase.instancedMissionAddress != null &&
_starbaseModel.currentBase.instancedMissionAddress != '')
inBattle = true;

return inBattle;
}

public function getCurrentUsersHomeSector():SectorData
{
return _starbaseModel.currentBase.sector;
}

public function hasSectorSplitTestCohort( sectorId:String ):Boolean
{
if (_sectorModel)
{
var sectors:Vector.<SectorVO> = _sectorModel.destinations;
var sector:SectorVO = null;
if (sectors)
{
for (var i:int = 0; i < sectors.length; ++i)
{
sector = sectors[i];
if (sector && sector.id == sectorId)
return
```

```
sector.splitTestCohortPrototype != null;
}
}
}
return false;
}

public function addOnPlayerVOAddedListener( callback:Function ):void {
_playerModel.onPlayerAdded.add(callback); }
public function removeOnPlayerVOAddedListener( callback:Function ):void {
_playerModel.onPlayerAdded.remove(callback); }

[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set chatController( v:ChatController ):void { _chatController = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set game( v:Game ):void { _game = v; }

override public function destroy():void
{
super.destroy();
}
}
}

---
File 391: igw\com\presenter\shared\TransitionPresenter.as
package com.presenter.shared
{
import com.Application;
import com.event.LoadEvent;
import com.event.ServerEvent;
import com.event.StateEvent;
import com.presenter.ImperiumPresenter;
import com.service.language.Localization;
import com.service.loading.ILoadService;

import
```

```
flash.events.Event;

import org.osflash.signals.Signal;

public class TransitionPresenter extends ImperiumPresenter implements ITransitionPresenter
{
private var _cleanupEvent:StateEvent;
private var _completeSignal:Signal;
private var _initEvent:StateEvent;
private var _loadService:ILoadService;
private var _updateSignal:Signal;
private var _failed:Boolean;

private var _creatingCharacter:String = 'CodeString.TransitionEvent.CreatingCharacter';
//Creating Character...
private var _connectingToProxy:String = 'CodeString.TransitionEvent.ConnectingToProxy';
//Connecting To Proxy...
private var _loggingIntoAccount:String = 'CodeString.TransitionEvent.LoggingIntoAccount';
//logging into account...
private var _connectingToBattle:String = 'CodeString.TransitionEvent.ConnectingToBattle';
//Connecting To Battle...
private var _connectingToSector:String = 'CodeString.TransitionEvent.ConnectingToSector';
//Connecting To Sector...
private var _connectingToStarbase:String =
'CodeString.TransitionEvent.ConnectingToStarbase'; //Connecting To Starbase...

[PostConstruct]
override public function init():void
{
_eventDispatcher.addEventListener(LoadEvent.LOCALIZATION_COMPLETE,
onLocalizationComplete);
super.init();
if (!_completeSignal)
_completeSignal = new Signal();
if (!_updateSignal)
_updateSignal = new Signal();
}

public function sendEvents():void
{
if (_cleanupEvent)
dispatch(_cleanupEvent);
if (_initEvent)
dispatch(_initEvent);
}

public function addEvents( initEvent:StateEvent, cleanupEvent:StateEvent ):void
{
_initEvent = initEvent;
_cleanupEvent = cleanupEvent;
}
```

```
public function transitionComplete():void { _completeSignal.dispatch(); }

public function addCompleteListener( callback:Function ):void { _completeSignal.add(callback); }
public function removeCompleteListener( callback:Function ):void {
_completeSignal.remove(callback); }

public function addUpdateListener( callback:Function ):void { _updateSignal.add(callback); }
public function removeUpdateListener( callback:Function ):void {
_updateSignal.remove(callback); }
public function updateView():void { _updateSignal.dispatch(); }

private function onLocalizationComplete( e:Event ):void { _updateSignal.dispatch(); }

public function get connectingText():String
{
if (!Localization.loaded)
return "...";
else if (Application.CONNECTION_STATE == ServerEvent.CONNECT_TO_PROXY)
return _connectingToProxy;
else if (Application.CONNECTION_STATE == ServerEvent.LOGIN_TO_ACCOUNT)
return _loggingIntoAccount;
else if (_initEvent == null)
return "...";
else if (_initEvent.type == StateEvent.GAME_BATTLE_INIT || _initEvent.type ==
StateEvent.GAME_BATTLE)
return _connectingToBattle;
else if (_initEvent.type == StateEvent.GAME_SECTOR_INIT || _initEvent.type ==
StateEvent.GAME_SECTOR)
return _connectingToSector;
return _connectingToStarbase;
}

public function get estimatedLoadCompleted():Number { return
_loadService.estimatedLoadCompleted; }

public function set failed( v:Boolean ):void { _failed = v; _updateSignal.dispatch()}
public function get failed():Boolean { return _failed; }

public function get trackAnalytics():Boolean
{
if (_initEvent == null) // || _initEvent.type == StateEvent.CONNECTING_TO_PROXY ||
_initEvent.type == StateEvent.LOGGING_INTO_ACCOUNT)
return false;
return true;
}

public function get hasWaiting():Boolean { return _loadService.highPrioritiesInProgress > 0; }

[Inject]
```

```
public function set loadService( v:ILoadService ):void { _loadService = v; }

override public function destroy():void
{
_eventDispatcher.removeEventListener(LoadEvent.LOCALIZATION_COMPLETE,
onLocalizationComplete);
super.destroy();
_completeSignal.removeAll();
_completeSignal = null;
_cleanupEvent = _initEvent = null;
_loadService.reset();
_loadService = null;
}

}
}

-----------------------------------
File 392: igw\com\presenter\shared\UIPresenter.as
package com.presenter.shared
{
import com.Application;
import com.controller.ChatController;
import com.controller.GameController;
import com.controller.ServerController;
import com.controller.SettingsController;
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.BattleEvent;
import com.event.SectorEvent;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.signal.InteractSignal;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.alliance.AllianceModel;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.battle.BattleModel;
import com.model.battle.BattleRerollVO;
import com.model.battlelog.BattleLogModel;
import com.model.blueprint.BlueprintModel;
import com.model.blueprint.BlueprintVO;
import com.model.event.EventModel;
import com.model.event.EventVO;
import com.model.mail.MailModel;
import com.model.mail.MailVO;
import
```

```
com.model.motd.MotDDailyRewardModel;
import com.model.motd.MotDModel;
import com.model.player.CurrentUser;
import com.model.player.PlayerModel;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.scene.SceneModel;
import com.model.sector.SectorModel;
import com.model.sector.SectorVO;
import com.model.starbase.BuffVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.model.transaction.TransactionVO;
import com.model.warfrontModel.WarfrontModel;
import com.presenter.ImperiumPresenter;
import com.service.loading.LoadPriority;
import com.service.server.outgoing.alliance.AllianceSendInviteRequest;

import flash.display.StageDisplayState;
import flash.utils.Dictionary;

import org.ash.core.Game;
import org.parade.core.IView;
import org.parade.core.ViewController;

public class UIPresenter extends ImperiumPresenter implements IUIPresenter
{
private static var _currentBookmarkCount:uint;

private var _currentState:String;

private var _game:Game;

private var _serverController:ServerController;
private var _chatController:ChatController;
private var _gameController:GameController;
private var _settingsController:SettingsController;
private var _viewController:ViewController;
private var _transactionController:TransactionController;

private var _sceneModel:SceneModel;
private var _sectorModel:SectorModel;
private var _battleLogModel:BattleLogModel;
private var _assetModel:AssetModel;
private var _starbaseModel:StarbaseModel;
private var _blueprintModel:BlueprintModel;
private var _motdModel:MotDModel;
private var _motdDailyModel:MotDDailyRewardModel;
private var _warfrontModel:WarfrontModel;
private var _battleModel:BattleModel;
private
```

```
var _transactionModel:TransactionModel;
private var _prototypeModel:PrototypeModel;
private var _mailModel:MailModel;
private var _allianceModel:AllianceModel;
private var _playerModel:PlayerModel;
private var _eventModel:EventModel;
private var _interactSignal:InteractSignal;
private var _starbaseFactory:IStarbaseFactory;

[PostConstruct]
override public function init():void
{
super.init();
}

public function gotoCoords( x:int, y:int, sector:String ):void
{

if (sector != _sectorModel.sectorID)
{
jumpToSector(x, y, sector);
} else
{
var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system != null)
system.moveToLocation(x, y);
else
{
jumpToSector(x, y, sector);
}
}

}

public function viewBattleReplay( battleKey:String ):void
{
var battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_REPLAY, battleKey);
dispatch(battleEvent);
}

public function linkCoords( x:int, y:int ):void
{
_chatController.linkCoords(x, y);
}

public function addBookmark( x:int, y:int ):void
{
if (_currentBookmarkCount == 0)
_currentBookmarkCount = CurrentUser.bookmarkCount + 1;

var
```

```
name:String = 'Bookmark ' + _currentBookmarkCount;
++_currentBookmarkCount;

var sectorName:String = _sectorModel.sectorName;
var sector:String = _sectorModel.sectorID;

var currentSector:SectorVO = _sectorModel.currentSectorVO;
CurrentUser.addBookmark(name, sector, currentSector.sectorPrototype,
currentSector.sectorNamePrototype, currentSector.sectorEnumPrototype, x, y,
CurrentUser.nextBookmarkIndex);
_gameController.bookmarkSave(name, sector, currentSector.sectorNamePrototype.name,
currentSector.sectorEnumPrototype.name, currentSector.sectorPrototype.name, x, y);
}

private function jumpToSector( x:int, y:int, sector:String ):void
{
var sectorEvent:SectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, sector, null,
x, y);
dispatch(sectorEvent);
}

public function changeResolution():void
{
_sceneModel.changeResolution();
_interactSignal.resolutionChange();
_interactSignal.scroll(0, 0);
}

public function toggleSFXMute():void
{
_settingsController.toggleSFXMute();
}

public function toggleMusicMute():void
{
_settingsController.toggleMusicMute();
}

public function setSFXVolume( v:Number ):void
{
_settingsController.setSFXVolume(v);
}

public function setMusicVolume( v:Number ):void
{
_settingsController.setMusicVolume(v);
}

public function toggleFullScreen():void
{
_settingsController.toggleFullScreen();
```

```
}

public function getTransactions():Dictionary
{
return _transactionModel.transactions;
}

public function loadIcon( url:String, callback:Function ):void
{
getFromCache("assets/" + url, callback);
}

public function loadIconFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
loadIcon("assets/" + _currentAssetVO.smallImage, callback);
}

public function loadMessageImage( url:String, callback:Function ):void
{
getFromCache(url, callback, LoadPriority.LOW, true);
}

public function loadPortraitSmall( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(getFromCache(portraitName));
getFromCache('assets/' + avatarVO.smallImage, callback);
}

public function loadPortraitMedium( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(getFromCache(portraitName));
getFromCache('assets/' + avatarVO.mediumImage, callback);
}

public function loadPortraitIcon( portraitName:String, callback:Function ):void
{
var avatarVO:AssetVO = AssetVO(getFromCache(portraitName));
getFromCache('assets/' + avatarVO.iconImage, callback);
}

public function getFromCache( url:String, callback:Function = null, priority:int =
LoadPriority.LOW, absoluteURL:Boolean = false ):Object
{
return _assetModel.getFromCache(url, callback, priority, absoluteURL);
}

public function getPrototypeUIIcon( prototype:IPrototype ):String
{
var
```

```
assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;

var currentAssetVO:AssetVO = _assetModel.getEntityData(assetName);
return currentAssetVO.iconImage;
}

public function getBlueprintByName( prototype:String ):BlueprintVO
{
return _blueprintModel.getBlueprintByName(prototype);
}

public function removeBlueprintByName( name:String ):void
{
_blueprintModel.removeBlueprintByName(name);
}

public function getBlueprintByID( id:String ):BlueprintVO
{
return _blueprintModel.getBlueprintByID(id);
}

public function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number
{
return _transactionController.getBlueprintHardCurrencyCost(blueprint, partsPurchased);
}

public function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void
{
_transactionController.buyBlueprintTransaction(blueprint, partsPurchased);
}
public function completeBlueprintResearch( blueprint:BlueprintVO):void
{
_transactionController.completeBlueprintResearchTransaction(blueprint);
}

public function purchaseReroll( battleKey:String ):void
{
_transactionController.starbasePurchaseReroll(battleKey);
}

public function purchaseDeepScan( battleKey:String ):void
{
_transactionController.starbasePurchaseDeepScan(battleKey);
}

public function addRerollFromRerollCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.add(callback);
```

```
}

public function removeRerollFromRerollCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.remove(callback);
}

public function addRerollFromScanCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.add(callback);
}

public function removeRerollFromScanCallback( callback:Function ):void
{
_battleModel.onRerollUpdated.remove(callback);
}

public function removeRerollFromAvailable( battleID:String ):void
{
_battleModel.removeRerollByID(battleID);
}

public function getPrototypeUIName( prototype:IPrototype ):String
{

var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;

var currentAssetVO:AssetVO = _assetModel.getEntityData(assetName);
return currentAssetVO.visibleName;
}

public function getAssetVOFromIPrototype( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function getAssetVO( assetName:String ):AssetVO
{
return _assetModel.getEntityData(assetName);
}

public function getBattleEndDialogByFaction( faction:String, combatResult:String = 'Victory'
):Vector.<IPrototype>
{
return
```

```
_prototypeModel.getBEDialogueByFaction(faction, combatResult);
}

public function getBlueprintPrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getBlueprintPrototype(name);
}

public function getFilterAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.getUnsafeValue('filterCategory');
return _assetModel.getEntityData(assetName);
}

public function getBuffPrototypes():Dictionary
{
var buffs:Dictionary = new Dictionary();
var v:Vector.<IPrototype> = _prototypeModel.getBuffPrototypes();
var l:uint = v.length;
for (var i:uint = 0; i < l; ++i)
{
var buffType:String = v[i].getValue('buffType');
if (!(buffType in buffs))
{
buffs[buffType] = v[i];
}
}
return buffs;
}

public function getBuffPrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getBuffPrototype(name);
}

public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getCommendationRankPrototypesByName( name:String ):IPrototype
{
return _prototypeModel.getCommendationRankPrototypesByName(name);
}

public function sendMailMessage( playerID:String, subject:String, body:String ):void
{
_gameController.mailSendMessage(playerID, subject, body);
}

public
```

```
function sendAllianceMailMessage( subject:String, body:String ):void
{
_gameController.mailSendAllianceMessage(subject, body);
}

public function getFAQPrototypes():Vector.<IPrototype>
{
return _prototypeModel.getFAQEntryPrototypes();
}

public function sendGetMailboxMessage():void
{
_gameController.mailGetMailbox();
}

public function getMailDetails( mailKey:String ):void
{
_gameController.mailGetMailDetail(mailKey);
}

public function deleteMail( v:Vector.<String> ):void
{
var len:uint = v.length;
for (var i:uint = 0; i < len; ++i)
{
_mailModel.deleteMail(v[i]);
}

_gameController.mailDelete(v);
}

public function mailRead( mailKey:String ):void
{
_mailModel.mailRead(mailKey);
}

public function getFactionPrototypesByName( name:String ):IPrototype
{
return _prototypeModel.getFactionPrototypeByName(name);
}

public function getRacePrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getRacePrototypeByName(name);
}

public function sendMotDMessageRead( key:String ):void
{
_gameController.requestMotDRead(key);
}

public
```

```
function requestPlayer( id:String, name:String = '' ):void
{
_gameController.leaderboardRequestPlayerProfile(id, name);
}

public function allianceSendInvite( playerKey:String ):void
{
var sendInvite:AllianceSendInviteRequest =
AllianceSendInviteRequest(_serverController.getRequest(ProtocolEnum.ALLIANCE_CLIENT,
RequestEnum.ALLIANCE_SEND_INVITE));
sendInvite.playerKey = playerKey;
_serverController.send(sendInvite);
}

public function sendDailyClaimRequest( header:int, protocolID:int ):void
{
_gameController.requestDailyClaim(header, protocolID);
}

public function getTransactionByID( id:String ):TransactionVO
{
return _transactionController.getBuffTransaction(id);
}

public function getOfferPrototypeByName( name:String ):IPrototype
{
return _prototypeModel.getOfferPrototypeByName(name);
}

public function getOfferItemsByItemGroup( itemGroup:String ):Vector.<IPrototype>
{
return _prototypeModel.getOfferItemsByItemGroup(itemGroup);
}

public function canEquip( prototype:IPrototype, slotType:String ):RequirementVO { return
_transactionController.canEquip(prototype, slotType); }

public function addBattleLogListUpdatedListener( callback:Function ):void {
_battleLogModel.battleLogListUpdated.add(callback); }
public function removeBattleLogListUpdatedListener( callback:Function ):void {
_battleLogModel.battleLogListUpdated.remove(callback); }

public function addBattleLogDetailUpdatedListener( callback:Function ):void {
_battleLogModel.battleLogDetailUpdated.add(callback); }
public function removeBattleLogDetailUpdatedListener( callback:Function ):void {
_battleLogModel.battleLogDetailUpdated.remove(callback); }

public function addMailCountUpdateListener( callback:Function ):void {
_mailModel.countUpdated.add(callback); }
public function removeMailCountUpdateListener( callback:Function ):void {
_mailModel.countUpdated.remove(callback);
```

```
}

public function addOnMailHeadersUpdatedListener( callback:Function ):void {
_mailModel.mailHeadersUpdated.add(callback); }
public function removeOnMailHeadersUpdatedListener( callback:Function ):void {
_mailModel.mailHeadersUpdated.remove(callback); }

public function addOnMailDetailUpdatedListener( callback:Function ):void {
_mailModel.mailDetailUpdated.add(callback); }
public function removeOnMailDetailUpdatedListener( callback:Function ):void {
_mailModel.mailDetailUpdated.remove(callback); }

public function addMotDUpdatedListener( callback:Function ):void {
_motdModel.newMessage.add(callback); }
public function removeMotDUpdatedListener( callback:Function ):void {
_motdModel.newMessage.remove(callback); }

public function addDailyRewardListener( callback:Function ):void {
_motdDailyModel.rewardResponse.add(callback); }
public function removeDailyRewardListener( callback:Function ):void {
_motdDailyModel.rewardResponse.remove(callback); }

public function addOnPlayerVOAddedListener( callback:Function ):void {
_playerModel.onPlayerAdded.add(callback); }
public function removeOnPlayerVOAddedListener( callback:Function ):void {
_playerModel.onPlayerAdded.remove(callback); }

public function addWarfrontUpdateListener( listener:Function ):void {
_warfrontModel.addUpdateListener(listener); }
public function removeWarfrontUpdateListener( listener:Function ):void {
_warfrontModel.removeUpdateListener(listener); }

public function addAvailableRerollUpdatedListener( callback:Function ):void {
_battleModel.onRerollAdded.add(callback); }
public function removeAvailableRerollUpdatedListener( callback:Function ):void {
_battleModel.onRerollAdded.remove(callback); }

public function addEventUpdatedListener( callback:Function ):void {
_eventModel.onEventsUpdated.add(callback); }
public function removeEventUpdatedListener( callback:Function ):void {
_eventModel.onEventsUpdated.remove(callback); }


override protected function onStateChange( e:StateEvent ):void
{
_currentState = e.type;
super.onStateChange(e);
}

public function fteNextStep():void
{
```

```
_fteController.nextStep();
}

public function fteSkip():void
{
_fteController.skipFTE();
}

public function get unreadMailCount():uint { return _mailModel.unreadCount; }
public function get mail():Vector.<MailVO> { return _mailModel.mail; }

public function getPrototypeByName( proto:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getBuildingPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getResearchPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getShipPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getStoreItemPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(proto);
return iproto;
}
public function getShipPrototypeByName( proto:String ):IPrototype
{
return _prototypeModel.getShipPrototype(proto);
}

public function getBattleLogList( filter:String ):void
{
_gameController.battleLogGetBattleList( filter );
}

public function getBattleLogDetails( battleKey:String ):void
{
_gameController.battleLogGetBattleDetail(battleKey);
}

public function getAvailableRerolls():Vector.<BattleRerollVO>
{
return _battleModel.getAllAvailableRerolls();
}

public function updateStarbasePlatform():void
{
if (Application.STATE == StateEvent.GAME_STARBASE)
_starbaseFactory.createStarbasePlatform(CurrentUser.id, true);
}

public
```

```
function getView( view:Class ):IView { return _viewController.getView(view); }

public function addTransactionListener( type:int, callback:Function ):void {
_transactionController.addListener(type, callback); }
public function removeTransactionListener( callback:Function ):void {
_transactionController.removeListener(callback); }

public function removeBuff( buff:BuffVO ):void { _starbaseModel.removeBuffByID(buff.id); }

public function get buffs():Vector.<BuffVO> { return _starbaseModel.currentBase.buffs; }
public function get bubbleTimeRemaining():Number { return _starbaseModel.currentBase ?
_starbaseModel.currentBase.bubbleTimeRemaining : 0; }

public function get isSFXMuted():Boolean { return _soundController.areSFXMuted; }
public function get isMusicMuted():Boolean { return _soundController.isMusicMuted; }
public function get sfxVolume():Number { return _soundController.sfxVolume; }
public function get musicVolume():Number { return _soundController.musicVolume; }
public function get isFullScreen():Boolean { return Application.STAGE.displayState ==
StageDisplayState.FULL_SCREEN_INTERACTIVE; }

public function get motdModel():MotDModel { return _motdModel; }

public function get motdDailyModel():MotDDailyRewardModel { return _motdDailyModel; }

public function get currentGameState():String { return _currentState; }

public function get igaContextMenuDefaultIndex():int { return
_sectorModel.igaContextMenuDefaultIndex; }
public function setIGAContextMenuDefaultIndex( v:int ):void {
_sectorModel.igaContextMenuDefaultIndex = v; }

public function get tyrContextMenuDefaultIndex():int { return
_sectorModel.tyrContextMenuDefaultIndex; }
public function setTYRContextMenuDefaultIndex( v:int ):void {
_sectorModel.tyrContextMenuDefaultIndex = v; }

public function get sovContextMenuDefaultIndex():int { return
_sectorModel.sovContextMenuDefaultIndex; }
public function setSOVContextMenuDefaultIndex( v:int ):void {
_sectorModel.sovContextMenuDefaultIndex = v; }

public function get csContextMenuDefaultIndex():int { return
_sectorModel.csContextMenuDefaultIndex; }
public function setCSContextMenuDefaultIndex( v:int ):void {
_sectorModel.csContextMenuDefaultIndex = v; }

public function get currentActiveEvent():EventVO { return _eventModel.currentActiveEvent; }

public function get activeEvents():Vector.<EventVO> { return _eventModel.activeEvents; }
public function get upcomingEvents():Vector.<EventVO> { return _eventModel.upcomingEvents;
}
```

```
[Inject]
public function set game( v:Game ):void { _game = v; }

[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set chatController( v:ChatController ):void { _chatController = v; }
[Inject]
public function set gameController( v:GameController ):void { _gameController = v; }
[Inject]
public function set settingsController( v:SettingsController ):void { _settingsController = v; }
[Inject]
public function set viewController( v:ViewController ):void { _viewController = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

[Inject]
public function set sceneModel( v:SceneModel ):void { _sceneModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set battleLogModel( v:BattleLogModel ):void { _battleLogModel = v; }
[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set blueprintModel( v:BlueprintModel ):void { _blueprintModel = v; }
[Inject]
public function set motdModel( v:MotDModel ):void { _motdModel = v; }
[Inject]
public function set motdDailyModel( v:MotDDailyRewardModel ):void { _motdDailyModel = v; }
[Inject]
public function set warfrontModel( v:WarfrontModel ):void { _warfrontModel = v; }
[Inject]
public function set battleModel( value:BattleModel ):void { _battleModel = value; }
[Inject]
public function set transactionModel( v:TransactionModel ):void { _transactionModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set mailModel( v:MailModel ):void { _mailModel = v; }
[Inject]
public function set allianceModel( v:AllianceModel ):void { _allianceModel = v; }
[Inject]
public function set playerModel( v:PlayerModel ):void { _playerModel = v; }
[Inject]
public
```

```
function set eventModel( v:EventModel ):void { _eventModel = v; }
[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }

[Inject]
public function set interactSignal( v:InteractSignal ):void { _interactSignal = v; }

override public function destroy():void
{
super.destroy();

_sceneModel = null;
_serverController;
_starbaseModel = null;
_interactSignal = null;
_warfrontModel = null;
_battleModel = null;
_starbaseFactory = null;
}
}
}
```

File 393: igw\com\presenter\starbase\AttackAlertPresenter.as

```
package com.presenter.starbase
{
import com.event.BattleEvent;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.sector.SectorModel;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;

import org.parade.core.ViewController;

public class AttackAlertPresenter extends ImperiumPresenter implements IAttackAlertPresenter
{
private var _fleetModel:FleetModel;
private var _sectorModel:SectorModel;
private var _starbaseModel:StarbaseModel;
private var _viewController:ViewController;

[PostConstruct]
override public function init():void
{
super.init();
}

public
```

```
function joinBattle( battleServerAddress:String, fleetID:String = null ):void
{
if (!hasBattleEnded(battleServerAddress, fleetID))
{
if (fleetID)
_sectorModel.focusFleetID = fleetID;
var event:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN, battleServerAddress);
dispatch(event);
}
}

public function removeAllAlerts( viewClass:Class ):void
{
_viewController.removeFromQueue(viewClass);
}

public function hasBattleEnded( battleServerAddress:String, fleetID:String = null ):Boolean
{
if (fleetID != null)
{
var fleetVO:FleetVO = _fleetModel.getFleet(fleetID);
if (fleetVO && fleetVO.battleServerAddress == battleServerAddress)
return false;
else
return true;
}
if (_starbaseModel.homeBase.battleServerAddress == battleServerAddress ||
_starbaseModel.homeBase.instancedMissionAddress == battleServerAddress ||
(_starbaseModel.centerSpaceBase && _starbaseModel.centerSpaceBase.battleServerAddress
== battleServerAddress))
return false;
return true;
}

public function addFleetUpdateListener( listener:Function ):void {
_fleetModel.onUpdatedFleetsSignal.add(listener); }
public function removeFleetUpdateListener( listener:Function ):void {
_fleetModel.onUpdatedFleetsSignal.remove(listener); }

[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set viewController( v:ViewController ):void { _viewController = v; }

override public function destroy():void
{
super.destroy();
```

```
_fleetModel = null;
_sectorModel = null;
_starbaseModel = null;
_viewController = null;
}
}
}

-----------------------------------
File 394: igw\com\presenter\starbase\ConstructionPresenter.as
package com.presenter.starbase
{
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.CurrencyEnum;
import com.enum.TypeEnum;
import com.enum.server.PurchaseTypeEnum;
import com.event.TransactionEvent;
import com.event.signal.TransactionSignal;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintModel;
import com.model.blueprint.BlueprintVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BaseVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionVO;
import com.presenter.ImperiumPresenter;

import flash.utils.Dictionary;

import org.ash.core.Game;
import org.parade.core.IView;
import org.parade.core.ViewController;

public class ConstructionPresenter extends ImperiumPresenter implements
IConstructionPresenter
{
private var _assetModel:AssetModel;
private var _blueprintModel:BlueprintModel;
private var _game:Game;
private var _prototypeModel:PrototypeModel;
private var _requirementsCache:Dictionary;
private
```

```
var _starbaseModel:StarbaseModel;
private var _starbaseFactory:IStarbaseFactory;
private var _transactionController:TransactionController;
private var _viewController:ViewController;
private var _working:Vector.<IPrototype>;

[PostConstruct]
override public function init():void
{
super.init();
_working = new Vector.<IPrototype>;
}

public function performTransaction( transactionType:String, prototype:IPrototype,
purchaseType:uint, ... args ):void
{
var buildingVO:BuildingVO;
switch (transactionType)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:

StarbaseInteractSystem(_game.getSystem(StarbaseInteractSystem)).buildFromPrototype(prototype,
purchaseType);
break;

case TransactionEvent.STARBASE_BUILDING_UPGRADE:
buildingVO = BuildingVO(prototype);
var upgradeVO:IPrototype =
_prototypeModel.getBuildingPrototype(buildingVO.getValue('upgrade'));
if (_transactionController.starbaseUpgradeBuilding(buildingVO, upgradeVO, purchaseType))
{
//if the upgrade is instant then we want to update the prototype
if (purchaseType == PurchaseTypeEnum.INSTANT)
{
buildingVO.prototype = upgradeVO;
}
}
break;

case TransactionEvent.STARBASE_BUILDING_RECYCLE:
//remove the building from the game
buildingVO = BuildingVO(prototype);
_starbaseModel.removeBuildingByID(buildingVO.id);

//deposit the recycle profits into the player's base
var baseVO:BaseVO = _starbaseModel.currentBase;
baseVO.deposit(Math.floor(buildingVO.alloyCost * 0.20), CurrencyEnum.ALLOY);
baseVO.deposit(Math.floor(buildingVO.creditsCost * 0.20), CurrencyEnum.CREDIT);
baseVO.deposit(Math.floor(buildingVO.energyCost * 0.20), CurrencyEnum.ENERGY);
baseVO.deposit(Math.floor(buildingVO.syntheticCost * 0.20), CurrencyEnum.SYNTHETIC);

if
```

```
(buildingVO.itemClass == TypeEnum.PYLON)
{
var starbaseSystem:StarbaseSystem = StarbaseSystem(_game.getSystem(StarbaseSystem));
starbaseSystem.findPylonConnections(_game.getEntity(buildingVO.id), true);
}

//send to the server
_transactionController.starbaseRecycleBuilding(buildingVO);
_starbaseFactory.destroyStarbaseItem(_game.getEntity(buildingVO.id));
break;

case TransactionEvent.STARBASE_RESEARCH:
_transactionController.starbaseStartResearch(prototype, purchaseType);
break;

default:
throw new Error("Unable to perform transaction of type " + transactionType);
break;
}
}

public function getComponents( groupID:String, subItemID:String, slotID:String,
showHighest:Boolean, showAdvancedOnly:Boolean, showCommonOnly:Boolean,
showUncommonOnly:Boolean, showRareOnly:Boolean, showEpicOnly:Boolean,
showLegendaryOnly:Boolean ):Vector.<IPrototype>
{
_working.length = 0;
_requirementsCache = new Dictionary(true);
var blueprint:BlueprintVO;
var module:IPrototype;
var modules:Vector.<IPrototype> = _prototypeModel.getModulesBySlotType(groupID);
for (var i:int = 0; i < modules.length; i++)
{
module = modules[i];
blueprint = _blueprintModel.getBlueprintByName(module.name);
if (module.getValue("filterCategory") == subItemID || (subItemID == "Blueprint" &&
module.getUnsafeValue('rarity') != "Common"))
{
if (canEquip(module, slotID).allMet)
{
if (showHighest)
{
if (_requirementsCache[module.itemClass] == null ||
_requirementsCache[module.itemClass].getValue("level") < module.getValue("level"))
{
if (_requirementsCache[module.itemClass] != null)
_working.splice(_working.indexOf(_requirementsCache[module.itemClass]), 1);
_requirementsCache[module.itemClass] = module;
_working.push(module);
}
}
```

```
else if(showAdvancedOnly)
{
if(module.getUnsafeValue('rarity') == "Advanced1" || module.getUnsafeValue('rarity') ==
"Advanced2" || module.getUnsafeValue('rarity') == "Advanced3")
{
_working.push(module);
}
}
else if(showCommonOnly)
{
if(module.getUnsafeValue('rarity') == "Common")
{
_working.push(module);
}
}
else if(showUncommonOnly)
{
if(module.getUnsafeValue('rarity') == "Uncommon")
{
_working.push(module);
}
}
else if(showRareOnly)
{
if(module.getUnsafeValue('rarity') == "Rare")
{
_working.push(module);
}
}
else if(showEpicOnly)
{
if(module.getUnsafeValue('rarity') == "Epic")
{
_working.push(module);
}
}
else if(showLegendaryOnly)
{
if(module.getUnsafeValue('rarity') == "Legendary")
{
_working.push(module);
}
}
else
_working.push(module);
}
}
}

if
```

```
(_working.length > 0)
_working.sort(orderComponents);
return _working;
}

public function canEquip( prototype:IPrototype, slotType:String ):RequirementVO { return
_transactionController.canEquip(prototype, slotType); }

protected function orderComponents( itemOne:IPrototype, itemTwo:IPrototype ):Number
{
if (!itemOne)
return -1;
if (!itemTwo)
return 1;

if (itemOne.getValue("sort") < itemTwo.getValue("sort"))
return -1;
else
return 1;
}

public function getResearchPrototypes( groupID:String, subItemID:String ):Vector.<IPrototype>
{
_working.length = 0;
_requirementsCache = new Dictionary(true);
/*var t:Number = getTimer();
trace('--------------------------------------------------');*/
var research:Vector.<IPrototype> =
_prototypeModel.getResearchPrototypesByBuilding(subItemID == "Blueprint" ? groupID :
groupID + subItemID);
if (research)
{
var blueprint:BlueprintVO;
var factionMet:Boolean;
var len:int = research.length;
var prototype:IPrototype;
var requirementMet:Boolean;
for (var i:int = 0; i < len; i++)
{
prototype = research[i];
// match research with blueprint by common key (uiAsset)
blueprint = _blueprintModel.getBlueprintByUIName(prototype.getValue('uiAsset'));
factionMet = (prototype.getValue('requiredFaction') == CurrentUser.faction ||
prototype.getValue('requiredFaction') == '');
if (factionMet && (prototype.getValue("filterCategory") == subItemID || (subItemID == "Blueprint"
&& blueprint != null)))
{
if (blueprint)
requirementMet =
getRequirementsBoolean(TransactionEvent.STARBASE_BLUEPRINT_PURCHASE, prototype);
else
```

```
requirementMet = getRequirementsBoolean(TransactionEvent.STARBASE_RESEARCH,
prototype);
if (requirementMet)
_working.push(prototype);
else if (!prototype.getValue('hideWhileLocked') || (blueprint && blueprint.partsCollected != 0 ||
blueprint && blueprint.complete))
_working.push(prototype);
}
}
}
if (_working.length > 0)
_working.sort(orderResearchItems);
//trace(getTimer() - t);
return _working;
}

public function getRequirementsBoolean( transactionType:String, prototype:IPrototype
):Boolean
{
if (_requirementsCache[prototype] != null)
return _requirementsCache[prototype];

var requirement:RequirementVO;
switch (transactionType)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
requirement = _transactionController.canBuild(prototype);
break;
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
requirement = _transactionController.canUpgrade(BuildingVO(prototype));
break;
case TransactionEvent.STARBASE_BLUEPRINT_PURCHASE:
case TransactionEvent.STARBASE_RESEARCH:
requirement = _transactionController.canPurchaseResearch(prototype);
break;
case TransactionEvent.STARBASE_REFIT_BUILDING:
requirement = _transactionController.canRefit(prototype);
break;
}
_requirementsCache[prototype] = requirement.allMet;
return requirement.allMet;
}

public function getRequirements( transactionType:String, prototype:IPrototype ):RequirementVO
{
var requirement:RequirementVO;
switch (transactionType)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
requirement
```

```
= _transactionController.canBuild(prototype);
break;
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
requirement = _transactionController.canUpgrade(BuildingVO(prototype));
break;
case TransactionEvent.STARBASE_BLUEPRINT_PURCHASE:
case TransactionEvent.STARBASE_RESEARCH:
requirement = _transactionController.canPurchaseResearch(prototype);
break;
case TransactionEvent.STARBASE_REFIT_BUILDING:
requirement = _transactionController.canRefit(prototype);
break;
}
return requirement;
}

private function orderResearchItems( itemOne:IPrototype, itemTwo:IPrototype ):Number
{
if (!itemOne)
return -1;

if (!itemTwo)
return 1;

var sortOne:Number = itemOne.getValue("sort");
var sortTwo:Number = itemTwo.getValue("sort");

var blueprintOne:BlueprintVO = _blueprintModel.getBlueprintByName(itemOne.name);
var isResearchedOne:Boolean = isResearched(itemOne.name);
var isLockedOne:Boolean = !((blueprintOne) ?
getRequirementsBoolean(TransactionEvent.STARBASE_BLUEPRINT_PURCHASE, itemOne) :
getRequirementsBoolean(TransactionEvent.STARBASE_RESEARCH, itemOne));

var blueprintTwo:BlueprintVO = _blueprintModel.getBlueprintByName(itemTwo.name);
var isResearchedTwo:Boolean = isResearched(itemTwo.name);
var isLockedTwo:Boolean = !((blueprintTwo) ?
getRequirementsBoolean(TransactionEvent.STARBASE_BLUEPRINT_PURCHASE, itemTwo) :
getRequirementsBoolean(TransactionEvent.STARBASE_RESEARCH, itemTwo));

if (blueprintOne && blueprintTwo)
{
if (!isResearchedOne && !isResearchedTwo)
{
if (sortOne < sortTwo)
return -1;
else if (sortOne > sortTwo)
return 1;
}
if (isResearchedOne && !isResearchedTwo)
return 1;
else
```

```
if (!isResearchedOne && isResearchedTwo)
return -1;
}

if (blueprintOne && !isResearchedOne && (isResearchedTwo || isLockedTwo))
return -1;
if (blueprintTwo && !isResearchedTwo && (isResearchedOne || isLockedOne))
return 1;

if (!isLockedOne && isLockedTwo)
return -1;
else if (isLockedOne && !isLockedTwo)
return 1;

if ((isLockedOne && !isResearchedOne) && (isResearchedTwo && !isLockedTwo))
return -1;
else if ((!isLockedOne && isResearchedOne) && (!isResearchedTwo && isLockedTwo))
return 1;

if (sortOne < sortTwo)
return -1;
else if (sortOne > sortTwo)
return 1;

return 0;
}

public function requirementsMet( proto:IPrototype ):Boolean
{
var requirementMet:Boolean;
var blueprint:BlueprintVO = _blueprintModel.getBlueprintByName(proto.name);

if (blueprint)
requirementMet =
getRequirementsBoolean(TransactionEvent.STARBASE_BLUEPRINT_PURCHASE, proto);
else
requirementMet = getRequirementsBoolean(TransactionEvent.STARBASE_RESEARCH,
proto);

return requirementMet;
}

public function isResearched( tech:String ):Boolean
{
if (tech != '')
{
var requiredBuildingClass:String =
_prototypeModel.getResearchPrototypeByName(tech).getValue('requiredBuildingClass');
return _transactionController.isResearched(tech, requiredBuildingClass);
} else
return
```

```
false;
}

public function loadImage( url:String, callback:Function ):void {
_assetModel.getFromCache("assets/" + url, callback); }

public function getAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function getBlueprint( name:String ):BlueprintVO { return
_blueprintModel.getBlueprintByName(name); }

public function getPrototypeByName( proto:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getBuildingPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getResearchPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getShipPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getStoreItemPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(proto);
return iproto;
}

public function getResearchItemPrototypeByName( v:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getShipPrototype(v);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(v);
return iproto;
}

public function getBuildingPrototypes( groupID:String, subItemID:String ):Vector.<IPrototype>
{
_working.length = 0;
var items:Vector.<IPrototype> = _prototypeModel.getBuildableBuildingPrototypes();
for (var i:int = 0; i < items.length; i++)
{
if (items[i].getValue("category") == groupID)
_working.push(items[i]);
}

if (_working.length > 0)
_working.sort(orderBuildings);
```

```
return _working;
}

public function getFilterNameByKey( v:String ):String
{
var assetVO:AssetVO = _assetModel.getEntityData(v);
if (assetVO)
return assetVO.visibleName;

return '';
}

public function getBuildingCount( buildingClass:String ):int { return
_starbaseModel.currentBase.getBuildingCount(buildingClass); }
public function getBuildingMaxCount( buildingClass:String ):int { return
_starbaseModel.currentBase.getBuildingMaxCount(buildingClass); }
public function getBuildingUpgrade( upgrade:String ):IPrototype { return
_prototypeModel.getBuildingPrototype(upgrade); }
public function getBuildingVO( id:String ):BuildingVO { return
_starbaseModel.getBuildingByID(id); }
public function getBuildingVOByClass( itemClass:String, highestLevel:Boolean = false
):BuildingVO { return _starbaseModel.getBuildingByClass(itemClass, highestLevel); }
public function getStarbaseBuildingTransaction( constructionCategory:String = null,
buildingID:String = null ):TransactionVO { return
_transactionController.getStarbaseBuildingTransaction(constructionCategory, buildingID); }
public function getStarbaseResearchTransaction( buildingType:String ):TransactionVO { return
_transactionController.getStarbaseResearchTransactionByBuildingType(buildingType); }

private function orderBuildings( itemOne:IPrototype, itemTwo:IPrototype ):Number
{
var assetVO:AssetVO = AssetModel.instance.getEntityData(itemOne.asset);
var nameOne:String = assetVO.visibleName;

assetVO = AssetModel.instance.getEntityData(itemTwo.asset);
var nameTwo:String = assetVO.visibleName;

if (nameOne < nameTwo)
{
return -1;
} else if (nameOne > nameTwo)
{
return 1;
} else
{
return 0;
}
}

public function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number
```

```
{ return _transactionController.getBlueprintHardCurrencyCost(blueprint, partsPurchased); }
public function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void {
_transactionController.buyBlueprintTransaction(blueprint, partsPurchased); }
public function completeBlueprintResearch( blueprint:BlueprintVO):void {
_transactionController.completeBlueprintResearchTransaction(blueprint); }

public function addOnTransactionRemovedListener( callback:Function ):void {
_transactionController.addListener(TransactionSignal.TRANSACTION_REMOVED, callback); }
public function removeOnTransactionRemovedListener( callback:Function ):void {
_transactionController.removeListener(callback); }

public function getView( view:Class ):IView { return _viewController.getView(view); }

public function mintNFT( tokenType:int, tokenAmount:int, tokenPrototype:String ):void
{
_transactionController.mintNFTTransaction(tokenType, tokenAmount, tokenPrototype);
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set blueprintModel( v:BlueprintModel ):void { _blueprintModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set viewController( v:ViewController ):void { _viewController = v; }

override public function destroy():void
{
super.destroy();

_assetModel = null;
_blueprintModel = null;
_game = null;
_prototypeModel = null;
_starbaseFactory = null;
_starbaseModel = null;
_transactionController = null;
_viewController = null;
_working.length = 0;
_working = null;
}
```

```
}
}

-----------------------------------
File 395: igw\com\presenter\starbase\FleetPresenter.as
package com.presenter.starbase
{
import com.Application;
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.TypeEnum;
import com.enum.server.PurchaseTypeEnum;
import com.event.BattleEvent;
import com.event.SectorEvent;
import com.event.StateEvent;
import com.event.signal.TransactionSignal;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionVO;
import com.presenter.ImperiumPresenter;

import org.ash.core.Entity;
import org.ash.core.Game;

public class FleetPresenter extends ImperiumPresenter implements IFleetPresenter
{
private static var _selectedFleetID:String;
private static var _shipSelectionFilter:Array;

private var _game:Game;
private var _assetModel:AssetModel;
private var _fleetModel:FleetModel;
private var _sectorModel:SectorModel;
private var _starbaseModel:StarbaseModel;
private var _prototypeModel:PrototypeModel;
private var _transactionController:TransactionController;

public function assignShipToFleet( selectedFleet:FleetVO, selectedShip:ShipVO, index:int ):void
{
_fleetModel.assignShipToFleet(selectedFleet, selectedShip, index);
_transactionController.dockUpdateFleet(selectedFleet);
```

```
}

public function removeShipFromFleet( selectedFleet:FleetVO, shipID:String ):void
{
_fleetModel.removeShipFromFleet(shipID);
_transactionController.dockUpdateFleet(selectedFleet);
}

public function changeFleetName( fleetToRename:FleetVO, newName:String ):void
{
_fleetModel.renameFleet(fleetToRename, newName);
_transactionController.dockChangeFleetName(fleetToRename.id, newName);
}

public function repairFleet( fleetToRepair:FleetVO, purchaseType:uint ):void
{

_transactionController.dockRepairShip(fleetToRepair, purchaseType);

if (purchaseType == PurchaseTypeEnum.INSTANT)
{
var id:String;
var fleetToUse:FleetVO;
id = fleetToRepair.id;
fleetToUse = fleetToRepair;
_fleetModel.repairFleet(fleetToUse, true);
}
}

public function updateRepair():void
{
var transaction:TransactionVO = _transactionController.getDockTransaction();
if (transaction && transaction.timeRemainingMS > 0)
_fleetModel.repairFleet(_fleetModel.getFleet(transaction.id), false, transaction);
}

public function cancelTransaction( transaction:TransactionVO ):void
{
_transactionController.transactionCancel(transaction.id);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache(url, callback);
}

public function loadIconFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
loadIcon("assets/"
```

```
+ _currentAssetVO.smallImage, callback);
}

public function getProtoTypeUIName( prototype:IPrototype, callback:Function ):void
{
var currentAssetVO:AssetVO = _assetModel.getEntityData(prototype.getValue('uiAsset'));
if (callback != null)
callback(currentAssetVO.visibleName);
}

public function launchFleet( fleetsToLaunch:Array ):void
{
_transactionController.dockLaunchFleet(fleetsToLaunch);

gotoFleet(fleetsToLaunch[0]);
}

public function gotoFleet( fleet:FleetVO ):void
{
var sectorEvent:SectorEvent;
if (fleet.inBattle)
{
_sectorModel.focusFleetID = fleet.id;
var battleEvent:BattleEvent = new BattleEvent(BattleEvent.BATTLE_JOIN,
fleet.battleServerAddress);
dispatch(battleEvent);
} else
{

if (Application.STATE == StateEvent.GAME_STARBASE)
{
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, fleet.sector != "" ? fleet.sector
: _starbaseModel.getBaseByID(fleet.starbaseID).sectorID, fleet.id);
dispatch(sectorEvent);
} else if (Application.STATE == StateEvent.GAME_SECTOR)
{

if (fleet.sector != '' && _sectorModel.sectorID != fleet.sector || (fleet.sector == '' &&
_sectorModel.sectorID != _starbaseModel.getBaseByID(fleet.starbaseID).sectorID))
{
sectorEvent = new SectorEvent(SectorEvent.CHANGE_SECTOR, fleet.sector != "" ? fleet.sector
: _starbaseModel.getBaseByID(fleet.starbaseID).sectorID, fleet.id);
dispatch(sectorEvent);
} else
{
var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system != null)

if (system != null)
{
```

```
var entity:Entity = _game.getEntity(fleet.id);
if (entity != null)
system.selectEntity(entity, true);
}
}

}
}
}

public function recallFleet( id:String ):void
{
if (id != '')
_transactionController.dockRecallFleet(id);
}

public function canRepair( fleet:FleetVO ):RequirementVO
{
return _transactionController.canRepair(fleet);
}

public function getAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function addTransactionListener( listener:Function ):void
{
_transactionController.addListener(TransactionSignal.TRANSACTION, listener);
_fleetModel.onUpdatedFleetsSignal.add(listener);
}

public function removeTransactionListener( listener:Function ):void
{
_transactionController.removeListener(listener);
_fleetModel.onUpdatedFleetsSignal.remove(listener);
}

public function addListenerOnFleetUpdated( listener:Function ):void
{
_fleetModel.onUpdatedFleetsSignal.add(listener);
}

public function removeListenerOnFleetUpdated( listener:Function ):void
{
_fleetModel.onUpdatedFleetsSignal.remove(listener);
}
```

```
public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getStatPrototypeByName( name:String ):IPrototype
{
var proto:IPrototype = _prototypeModel.getStatPrototypeByName(name);
return proto;
}

public function getFleet( id:String ):FleetVO
{
return _fleetModel.getFleet(id);
}

public function get dockLevel():int { return
BuildingVO(_starbaseModel.getBuildingByClass(TypeEnum.DOCK)).level; }
public function get dockTransaction():TransactionVO { return
_transactionController.getDockTransaction(); }
public function get shipyardTransaction():TransactionVO { return
_transactionController.getShipyardTransaction(); }
public function get fleets():Vector.<FleetVO> { return _fleetModel.fleets; }
public function get maxFleetPower():int { return _starbaseModel.currentBase.maxPower; }
public function get unassignedShips():Vector.<ShipVO> { return _fleetModel.ships; }
public function set selectedFleetID( v:String ):void { _selectedFleetID = v; }
public function get selectedFleetID():String { return _selectedFleetID; }
public function set shipSelectionFilter( v:Array ):void { _shipSelectionFilter = v; }
public function get shipSelectionFilter():Array { return _shipSelectionFilter; }

[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
}
}
```

File 396: igw\com\presenter\starbase\IAttackAlertPresenter.as

```
package com.presenter.starbase
{
import com.presenter.IImperiumPresenter;

public interface IAttackAlertPresenter extends IImperiumPresenter
{
function joinBattle( battleServerAddress:String, fleetID:String = null ):void;
function removeAllAlerts( viewClass:Class ):void
function hasBattleEnded( battleServerAddress:String, fleetID:String = null ):Boolean;
function addFleetUpdateListener( listener:Function ):void;
function removeFleetUpdateListener( listener:Function ):void;
}
}
```

File 397: igw\com\presenter\starbase\IConstructionPresenter.as

```
package com.presenter.starbase
{
import com.controller.transaction.requirements.RequirementVO;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.transaction.TransactionVO;
import com.presenter.IImperiumPresenter;

import org.parade.core.IView;

public interface IConstructionPresenter extends IImperiumPresenter
{
function performTransaction( transactionType:String, prototype:IPrototype, purchaseType:uint,
... args ):void;
function getBuildingPrototypes( groupID:String, subItemID:String ):Vector.<IPrototype>;
function getBuildingCount( buildingClass:String ):int;
function getBuildingMaxCount( buildingClass:String ):int;
function getBuildingUpgrade( upgrade:String ):IPrototype;
function getBuildingVO( id:String ):BuildingVO;
function getBuildingVOByClass( itemClass:String, highestLevel:Boolean = false ):BuildingVO;
function getStarbaseBuildingTransaction( constructionCategory:String = null, buildingID:String =
null ):TransactionVO;
function getStarbaseResearchTransaction( buildingType:String ):TransactionVO;

function getComponents( groupID:String, subItemID:String, slotID:String, showHighest:Boolean,
showAdvancedOnly:Boolean, showCommonOnly:Boolean, showUncommonOnly:Boolean,
showRareOnly:Boolean, showEpicOnly:Boolean,
showLegendaryOnly:Boolean):Vector.<IPrototype>;
function canEquip( prototype:IPrototype, slotType:String ):RequirementVO;

function
```

```
getResearchPrototypes( groupID:String, subItemID:String ):Vector.<IPrototype>;

function isResearched( tech:String ):Boolean;
function getRequirements( transactionType:String, prototype:IPrototype ):RequirementVO;
function requirementsMet( proto:IPrototype ):Boolean;
function loadImage( url:String, callback:Function ):void;
function getAssetVO( prototype:IPrototype ):AssetVO;
function getBlueprint( name:String ):BlueprintVO;
function getPrototypeByName( proto:String ):IPrototype;
function getResearchItemPrototypeByName( v:String ):IPrototype;
function getFilterNameByKey( v:String ):String;

function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number;
function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void;
function completeBlueprintResearch( blueprint:BlueprintVO):void;

function addOnTransactionRemovedListener( callback:Function ):void;
function removeOnTransactionRemovedListener( callback:Function ):void;

function getView( view:Class ):IView;
function mintNFT( tokenType:int, tokenAmount:int, tokenPrototype:String ):void;
}
}

-----------------------------------
File 398: igw\com\presenter\starbase\IFleetPresenter.as
package com.presenter.starbase
{
import com.controller.transaction.requirements.RequirementVO;
import com.model.asset.AssetVO;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.transaction.TransactionVO;
import com.presenter.IImperiumPresenter;

public interface IFleetPresenter extends IImperiumPresenter
{
function assignShipToFleet( selectedFleet:FleetVO, selectedShip:ShipVO, index:int ):void;
function removeShipFromFleet( selectedFleet:FleetVO, shipID:String ):void;
function changeFleetName( fleetToRename:FleetVO, newName:String ):void
function repairFleet( fleetToRepair:FleetVO, purchaseType:uint ):void;
function updateRepair():void;
function cancelTransaction( transaction:TransactionVO ):void;
function loadIcon( url:String, callback:Function ):void
function loadIconFromEntityData( type:String, callback:Function ):void;
function getProtoTypeUIName( prototype:IPrototype, callback:Function ):void;
function getAssetVO( prototype:IPrototype ):AssetVO;
function launchFleet( fleetsToLaunch:Array ):void;
function
```

```
gotoFleet( fleet:FleetVO ):void;
function recallFleet( id:String ):void;
function canRepair( fleet:FleetVO ):RequirementVO;
function addTransactionListener( listener:Function ):void;
function removeTransactionListener( listener:Function ):void;
function getStatPrototypeByName( name:String ):IPrototype;
function getConstantPrototypeValueByName( name:String ):Number;
function getFleet( id:String ):FleetVO;
function addListenerOnFleetUpdated( listener:Function ):void;
function removeListenerOnFleetUpdated( listener:Function ):void;

function get dockLevel():int;
function get dockTransaction():TransactionVO;
function get shipyardTransaction():TransactionVO;
function get fleets():Vector.<FleetVO>;
function get maxFleetPower():int;
function get unassignedShips():Vector.<ShipVO>;
function set selectedFleetID( v:String ):void;
function get selectedFleetID():String;
function set shipSelectionFilter( v:Array ):void;
function get shipSelectionFilter():Array;
}
}



-----------------------------------
File 399: igw\com\presenter\starbase\IMissionPresenter.as
package com.presenter.starbase
{
import com.model.mission.MissionInfoVO;
import com.model.mission.MissionVO;
import com.presenter.IImperiumPresenter;

public interface IMissionPresenter extends IImperiumPresenter
{
function getMissionInfo( type:String, chapterID:int = -1, missionID:int = -1,
forCaptainsLog:Boolean = false ):MissionInfoVO;
function acceptMission():void;
function showReward():void;
function showSector():void;
function acceptMissionReward():void;
function dispatchMissionEvent( type:String ):void;
function loadIcon( url:String, callback:Function ):void;
function fteNextStep():void;
function fteSkip():void;
function startInstancedMission(id:String):void;
function isInstancedMissionOn():Boolean;
function moveToMissionTarget():String;
function getStoryMission( chapter:int, mission:int ):MissionVO;
function
```

```
unpauseBattle():void;
function addTransactionListener( callback:Function ):void;
function removeTransactionListener( callback:Function ):void;
function requestAllScores():void;
function onAddAllScoresUpdatedListener( Listener:Function ):void;
function onRemoveAllScoresUpdatedListener( Listener:Function ):void;

function get currentMission():MissionVO;
function get fteStep():int;
}
}



-----------------------------------
File 400: igw\com\presenter\starbase\IShipyardPresenter.as
package com.presenter.starbase
{
import com.controller.transaction.requirements.RequirementVO;
import com.model.asset.AssetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.transaction.TransactionVO;
import com.presenter.IImperiumPresenter;

public interface IShipyardPresenter extends IImperiumPresenter
{
function loadIcon( url:String, callback:Function ):void;
function loadIconFromEntityData( type:String, callback:Function ):void;
function getModules( slotType:String ):Vector.<IPrototype>;
function getConstantPrototypeValueByName( name:String ):Number
function getSlotPrototype( key:String ):IPrototype;
function getBuildingShip():ShipVO;
function getShipByID( id:String ):ShipVO;
function buildShip( ship:ShipVO, purchaseType:uint ):void;
function refitShip( ship:ShipVO, purchaseType:uint ):void;
function cancelTransaction( transaction:TransactionVO ):void;
function recycleShip( shipVO:ShipVO ):void
function getAssetVO( prototype:IPrototype ):AssetVO;
function isResearched( tech:String ):Boolean;
function addTransactionListener( listener:Function ):void;
function removeTransactionListener( listener:Function ):void;
function canBuild( ship:IPrototype ):RequirementVO;
function isShipyardRepairing():Boolean;
function getPrototypeByName( proto:String ):IPrototype;
function get currentShip():ShipVO;
function set currentShip( v:ShipVO ):void;
function get refittingShip():ShipVO;
function get savedShip():ShipVO;
function set savedShip( v:ShipVO ):void;
function
```

```
get shipPrototypes():Vector.<IPrototype>;
function get shipyardTransaction():TransactionVO;
function get builtShipCount():Number;
function get maxAvailableShipSlots():Number;
function get canBuildNewShips():Boolean;
function getStatPrototypeByName( name:String ):IPrototype;
}
}



-----------------------------------
File 401: igw\com\presenter\starbase\IStarbasePresenter.as
package com.presenter.starbase
{
import com.controller.transaction.requirements.RequirementVO;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BaseVO;
import com.model.starbase.BuildingVO;
import com.model.transaction.TransactionVO;
import com.presenter.shared.IGamePresenter;

import org.ash.core.Entity;

public interface IStarbasePresenter extends IGamePresenter
{
function cancelTransaction( trans:TransactionVO ):void;
function moveEntity():void;
function onInteractionWithBaseEntity( x:int, y:int, baseEntity:Entity ):void;
function performTransaction( transactionType:String, prototype:IPrototype, purchaseType:uint,
... args ):void;

function getBuildingUpgrade( buildingVO:BuildingVO ):IPrototype;
function getBuildingVO( id:String ):BuildingVO;
function getBuildingVOByClass( itemClass:String, highestLevel:Boolean = false ):BuildingVO;
function getPrototypeByName( proto:String ):IPrototype;
function getBlueprintByName( prototype:String ):BlueprintVO;
function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number;
function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void;
function getRequirements( transactionType:String, prototype:IPrototype ):RequirementVO;
function getSlotType( key:String ):String;
function getFilterAssetVO( prototype:IPrototype ):AssetVO;
function getStarbaseBuildingTransaction( constructionCategory:String = null, buildingID:String =
null ):TransactionVO;
function getStarbaseResearchTransaction( buildingType:String ):TransactionVO;
function loadIcon( url:String, callback:Function ):void;
function getRepairCost():int;
function
```

```
getRepairTime( getTotal:Boolean = false ):int;
function getEntityName( assetName:String ):String;

function showBuildings():void;

function addBaseInteractionListener( callback:Function ):void;
function addTransactionListener( listener:Function ):void;
function removeTransactionListener( listener:Function ):void;
function addOnGenericAllianceMessageRecievedListener( callback:Function ):void;
function removeOnGenericAllianceMessageRecievedListener( callback:Function ):void;

function getConstantPrototypeValueByName( name:String ):Number;
function getStatPrototypeByName( name:String ):IPrototype;

function get buildingPrototypes():Vector.<IPrototype>;
function get currentBase():BaseVO;
function get researchPrototypes():Vector.<IPrototype>;

function get totalDamagedBuildings():int;
function get totalDestroyedBuildings():int;
function get totalBaseDamage():Number;
}
}

-------------------------------------
File 402: igw\com\presenter\starbase\IStorePresenter.as
package com.presenter.starbase
{

import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.presenter.IImperiumPresenter;

import flash.utils.Dictionary;

public interface IStorePresenter extends IImperiumPresenter
{
function getTransactions():Dictionary;
function getStoreItemPrototypes():Vector.<IPrototype>;
function getCurrentState():String;
function getPrototypeUIName( prototype:IPrototype ):String;
function getPrototypeUISmallImage( prototype:IPrototype ):String;
function getProtoTypeUIDescriptionText( prototype:IPrototype ):String;
function loadIconFromPrototype( prototype:IPrototype, callback:Function ):void
function addOnTransactionUpdatedListener( callback:Function ):void;
function removeOnTransactionUpdatedListener( callback:Function ):void;
function addOnTransactionRemovedListener( callback:Function ):void;
function
```

```
removeOnTransactionRemovedListener( callback:Function ):void;
function getHardCurrencyCostFromSeconds( buildTimeSeconds:Number ):int;
function getHardCurrencyCostFromResource( resources:int, type:String ):int;
function getCanAfford( prototype:IPrototype ):Boolean;
function getShipById( id:String ):ShipVO;
function getFleetById( id:String ):FleetVO;
function getBuildingByID( id:String ):BuildingVO;
function getResearchByID( id:String ):ResearchVO;
function getBuildingVOByClass( itemClass:String ):BuildingVO;
function getMaxResources():uint;
function getMaxCredits():uint;
function getResourceCount( type:String ):uint;
function speedUpTransaction( serverKey:String, token:int, instant:Boolean, speedUpBy:int,
fromStore:Boolean, cost:int ):void;
function buyResourceTransaction( prototype:IPrototype, percent:int, centerBase:Boolean,
cost:int ):void;
function buyItemTransaction( buffPrototype:IPrototype, centerBase:Boolean, cost:int ):void;
function buyOtherItemTransaction( prototype:IPrototype, amount:int, centerBase:Boolean,
cost:int ):void;
}
}

------------------------------------
File 403: igw\com\presenter\starbase\ITradePresenter.as
package com.presenter.starbase
{
import com.model.prototype.IPrototype;
import com.model.starbase.TradeRouteVO;
import com.model.transaction.TransactionVO;
import com.presenter.IImperiumPresenter;

public interface ITradePresenter extends IImperiumPresenter
{
function getConstantPrototypeValueByName( name:String ):Number;
function getProtoTypeUIName( prototype:IPrototype ):String;
function getPrototypeUIDescription( prototype:IPrototype ):String;
function loadIconFromPrototype( type:String, prototype:IPrototype, callback:Function ):void;
function cancelContract( id:String ):void;
function requestContract( centerSpaceBase:Boolean, contractPrototype:String,
factionPrototype:String, callback:Function ):void;
function getContractsFromFaction( contractGroup:String ):Vector.<IPrototype>;
function getBalancedContractFromFaction( contractGroup:String ):IPrototype;
function getAgentsFromFaction( contractGroup:String ):Vector.<IPrototype>;
function getAgent( contractGroup:String, reputation:Number ):IPrototype;
function getAgentDialogByGroup( name:String ):Vector.<IPrototype>;
function hasAgentGreetingBeenViewed( agentID:int ):Boolean;
function setAgentGreetingViewed( agentID:int ):void;
function getTradeRouteTransaction( id:String ):TransactionVO;
function addTransactionListener( callback:Function ):void;
function removeTransactionListener( callback:Function ):void;

function
```

```
get tradeRouteCreditIncome():uint;
function get tradeRouteResourceIncome():uint;
function get maxContracts():int;
function get maxUnlockedContracts():int;
function get tradeRoutes():Vector.<TradeRouteVO>;
}
}



---
File 404: igw\com\presenter\starbase\MissionPresenter.as
package com.presenter.starbase
{
import com.Application;
import com.controller.ServerController;
import com.controller.fte.FTEController;
import com.controller.transaction.TransactionController;
import com.enum.TypeEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.event.MissionEvent;
import com.event.SectorEvent;
import com.event.StarbaseEvent;
import com.event.TransitionEvent;
import com.event.StateEvent;
import com.event.signal.TransactionSignal;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Position;
import com.game.entity.nodes.shared.grid.GridNode;
import com.game.entity.systems.interact.SectorInteractSystem;
import com.game.entity.systems.shared.grid.GridSystem;
import com.model.achievements.AchievementModel;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.mission.MissionInfoVO;
import com.model.mission.MissionModel;
import com.model.mission.MissionVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.sector.SectorModel;
import com.model.starbase.StarbaseModel;
import com.presenter.ImperiumPresenter;
import com.service.language.Localization;
import com.service.server.outgoing.battle.BattlePauseRequest;

import flash.events.Event;

import
```

```
org.ash.core.Entity;
import org.ash.core.Game;
import org.shared.ObjectPool;


public class MissionPresenter extends ImperiumPresenter implements IMissionPresenter
{
private var _assetModel:AssetModel;
private var _fleetModel:FleetModel;
private var _game:Game;
private var _locToken:Object;
private var _missionModel:MissionModel;
private var _achievementModel:AchievementModel;
private var _prototypeModel:PrototypeModel;
private var _sectorModel:SectorModel;
private var _serverController:ServerController;
private var _starbaseModel:StarbaseModel;
private var _transactionController:TransactionController;

private var _launchFleetForMission:String = 'CodeString.Toast.LaunchFleetForMission';
private var _returnFleetToHomeSectorForMission:String =
'CodeString.Toast.ReturnFleetToHomeSectorForMission';

[PostConstruct]
override public function init():void
{
super.init();
_locToken = {"[[String.PlayerName]]":CurrentUser.name};
}

public function getMissionInfo( type:String, chapterID:int = -1, missionID:int = -1,
forCaptainsLog:Boolean = false ):MissionInfoVO
{
var dialog:IPrototype;
var info:MissionInfoVO = ObjectPool.get(MissionInfoVO);
var mission:MissionVO = (chapterID == -1 || missionID == -1) ? _missionModel.currentMission :
_missionModel.getStoryMission(chapterID, missionID);
if (!mission)
return null;
var npc:IPrototype;
var progress:int = 0;


switch (type)
{
case MissionEvent.MISSION_FAILED:
dialog = _prototypeModel.getDialogPrototypeByName(mission.failDialogue);
info.addDialog(Localization.instance.getStringWithTokens(dialog.getValue('dialogString'),
_locToken));
//todo uncomment when ready
//var
```

```
audioDir:String = dialog.getValue('dialogAudioString');
//if(audioDir.length>0)
// info.addSound(audioDir);
//info.addSound("sounds/sfx/AFX_Base_Weapon_Missle_Pod_v001A.mp3");
addSpeaker(dialog, info);
progress = 2;
break;
case MissionEvent.MISSION_GREETING:
dialog = _prototypeModel.getDialogPrototypeByName(mission.greetingDialogue);
info.addDialog(Localization.instance.getStringWithTokens(dialog.getValue('dialogString'),
_locToken));
//todo uncomment when ready
//var audioDir:String = dialog.getValue('dialogAudioString');
//if(audioDir.length>0)
// info.addSound(audioDir);
//info.addSound("sounds/sfx/AFX_Base_Weapon_Missle_Pod_v001A.mp3");
addSpeaker(dialog, info);
dialog = _prototypeModel.getDialogPrototypeByName(mission.briefingDialogue);
info.addDialog(Localization.instance.getStringWithTokens(dialog.getValue('dialogString'),
_locToken));
//todo uncomment when ready
//var audioDir:String = dialog.getValue('dialogAudioString');
//if(audioDir.length>0)
// info.addSound(audioDir);
//info.addSound("sounds/sfx/AFX_Base_Weapon_Missle_Pod_v001A.mp3");
addSpeaker(dialog, info);
progress = 0;
//for non kill missions we also want to add the situational dialog into the missionInfoVO
if (forCaptainsLog || mission.progressEvent == "Kill")
break;
case MissionEvent.MISSION_SITUATIONAL:
dialog = _prototypeModel.getDialogPrototypeByName(mission.situationDialogue);
info.addDialog(Localization.instance.getStringWithTokens(dialog.getValue('dialogString'),
_locToken));
//todo uncomment when ready
//var audioDir:String = dialog.getValue('dialogAudioString');
//if(audioDir.length>0)
// info.addSound(audioDir);
//info.addSound("sounds/sfx/AFX_Base_Weapon_Missle_Pod_v001A.mp3");
addSpeaker(dialog, info);
progress = (mission.progressEvent == "Kill") ? 2 : 0;
break;
case MissionEvent.MISSION_VICTORY:
dialog = _prototypeModel.getDialogPrototypeByName(mission.victoryDialogue);
info.addDialog(Localization.instance.getStringWithTokens(dialog.getValue('dialogString'),
_locToken));

//todo uncomment when ready
//var audioDir:String = dialog.getValue('dialogAudioString');
//if(audioDir.length>0)
//
```

```
info.addSound(audioDir);
//info.addSound("sounds/sfx/AFX_Base_Weapon_Missle_Pod_v001A.mp3");
addSpeaker(dialog, info);
progress = 3;
break;
}


//objectives
var objectives:Array = mission.objectives.split(',');
var objProto:IPrototype;
for (var i:int = 0; i < objectives.length; i++)
{
objProto = _prototypeModel.getMissionObjective(objectives[i]);
if (objProto)
info.addObjective(Localization.instance.getString(objProto.getValue('dialogString')));
}

//rewards
info.alloyReward = mission.alloyReward;
info.creditReward = mission.creditsReward;
info.energyReward = mission.energyReward;
info.syntheticReward = mission.syntheticReward;
info.palladiumCurrencyReward = mission.palladiumCurrencyReward;
info.blueprintReward = mission.blueprintReward;

info.currentProgress = progress; //mission.progress;
info.progressRequired = 3; //mission.progressRequired;

//object pool the missionVO if we had to make a new one to get the info
if (chapterID != -1 && missionID != -1 && (chapterID != _missionModel.currentMission.chapter ||
missionID != _missionModel.currentMission.mission))
ObjectPool.give(mission);

return info;
}

public function acceptMission():void
{
var mission:MissionVO = _missionModel.currentMission;
_missionModel.missionAccepted();
_transactionController.missionAccept(mission.id);
}
public function startInstancedMission(id:String):void
{
_transactionController.instancedMissionStart(id);

var event:TransitionEvent = new TransitionEvent(TransitionEvent.TRANSITION_BEGIN);
dispatch(event);
}
public
```

```
function isInstancedMissionOn():Boolean
{
if(_starbaseModel.homeBase.instancedMissionAddress == null)
return false;
else
return true;
}

public function showReward():void
{
var missionEvent:MissionEvent = new MissionEvent(MissionEvent.SHOW_REWARDS);
dispatch(missionEvent);
}

public function acceptMissionReward():void
{
var mission:MissionVO = _missionModel.currentMission;
_missionModel.missionRewardAccepted();
_transactionController.missionAcceptRewards(mission.id);
}

public function dispatchMissionEvent( type:String ):void
{
var missionEvent:MissionEvent = new MissionEvent(type);
dispatch(missionEvent);
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function fteNextStep():void
{
if (_fteController.running)
_fteController.nextStep();
}

public function showSector():void
{
var event:Event;
event = new SectorEvent(SectorEvent.CHANGE_SECTOR,
_starbaseModel.homeBase.sectorID);
dispatch(event);
}
public function fteSkip():void
{
if (_fteController.running)
_fteController.skipFTE();
}

private
```

```
function findClosestTransgate( target:Entity ):Entity
{
var entity:Entity;
var bestDistSq:Number = 0;
var targetPos:Position = target.get(Position);

// This will only work in sector mode, because it's the only place transgates are found, naturally.
var detail:Detail;
var pos:Position;
var distSq:Number;
var gridSystem:GridSystem = GridSystem(_game.getSystem(GridSystem));
for (var node:GridNode = gridSystem.nodes.head; node; node = node.next)
{
detail = node.entity.get(Detail);
switch (detail.type)
{
case TypeEnum.TRANSGATE_IGA:
case TypeEnum.TRANSGATE_SOVEREIGNTY:
case TypeEnum.TRANSGATE_TYRANNAR:
break;
default:
continue;
}

pos = node.entity.get(Position);
distSq = (pos.x - targetPos.x) * (pos.x - targetPos.x) + (pos.y - targetPos.y) * (pos.y -
targetPos.y);
if (bestDistSq <= 0 || distSq < bestDistSq)
{
entity = node.entity;
bestDistSq = distSq;
}
}

return entity;
}

public function moveToMissionTarget():String
{
var mission:MissionVO = currentMission;
//send the player back to their base if they're doing a non-kill mission
if (mission.progressEvent != "Kill")
{
if (Application.STATE != StateEvent.GAME_STARBASE)
{
var starbaseEvent:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
dispatch(starbaseEvent);
}
return null;
}

var
```

```
fleet:FleetVO;
var fleets:Vector.<FleetVO> = _fleetModel.fleets;
var activeFleet:FleetVO;
var hasFleetLaunched:Boolean = false;
for (var i:int = 0; i < fleets.length; i++)
{
fleet = fleets[i];
if (fleet.sector != "")
{
hasFleetLaunched = true;
}
}
if(!hasFleetLaunched)
{
return _launchFleetForMission;
}

var activeFleetId:String = _sectorModel.focusFleetID;
activeFleet = _fleetModel.getFleet(activeFleetId);

//TODO: handle the case when no fleet is active but still launched (e.g., recently selected fleet is
not in the current sector)
if(!activeFleet)
return null;

var system:SectorInteractSystem =
SectorInteractSystem(_game.getSystem(SectorInteractSystem));
if (system)
{
//are we in the same sector as the mission entity?
if (mission.sector == _sectorModel.sectorID)
{
var missionEntities:Vector.<Entity> = system.missionEntities;
if (missionEntities.length > 0 && activeFleet)
{
var entity:Entity = missionEntities[0];
var position:Position = entity.get(Position);
system.moveToLocation(position.x, position.y, 1.3);
return null;
}
} else
{
//we're not in the same sector so point to a nearby transgate we can travel through
if (activeFleet)
{
var fleetEntity:Entity = _game.getEntity(activeFleetId);
if (fleetEntity)
{
var closestTransgate:Entity = findClosestTransgate(fleetEntity);
if (closestTransgate)
{
```

```
var transgatePos:Position = closestTransgate.get(Position);
system.moveToLocation(transgatePos.x, transgatePos.y, 1.3);
}
}
return null;
}
}
return _launchFleetForMission;
}

var event:SectorEvent;
if (activeFleet)
event = new SectorEvent(SectorEvent.CHANGE_SECTOR, activeFleet.sector, activeFleet.id);
if (event)
{
dispatch(event);
return null;
}

if (Application.STATE == StateEvent.GAME_SECTOR)
{
if (hasFleetLaunched &&
mission.sector != _sectorModel.sectorID &&
_starbaseModel.homeBase.sectorID != _sectorModel.sectorID)
{
return _returnFleetToHomeSectorForMission;
}
}

return _launchFleetForMission;
}

public function getStoryMission( chapter:int, mission:int ):MissionVO
{
return _missionModel.getStoryMission(chapter, mission);
}

public function unpauseBattle():void
{
var pauseRequest:BattlePauseRequest =
BattlePauseRequest(_serverController.getRequest(ProtocolEnum.BATTLE_CLIENT,
RequestEnum.BATTLE_PAUSE));
pauseRequest.pause = false;
_serverController.send(pauseRequest);
}

public function requestAllScores():void
{
_serverController.requestAllScores();
}
```

```
public function onAddAllScoresUpdatedListener( Listener:Function ):void {
_achievementModel.onAllScoresUpdated.add(Listener); }
public function onRemoveAllScoresUpdatedListener( Listener:Function ):void {
_achievementModel.onAllScoresUpdated.remove(Listener); }

public function addTransactionListener( callback:Function ):void {
_transactionController.addListener(TransactionSignal.DATA_IMPORTED, callback); }
public function removeTransactionListener( callback:Function ):void {
_transactionController.removeListener(callback); }

private function addSpeaker( dialog:IPrototype, info:MissionInfoVO ):void
{
//set the npc and title
var assetVO:AssetVO;
var key:String = dialog.getValue('speaker').replace(/ /g, '');
var npc:IPrototype = _prototypeModel.getNPCPrototypeByName(key);
if (npc)
assetVO = _assetModel.getEntityData(npc.getValue("race"));
if (assetVO)
{
info.addTitle(assetVO.visibleName, 0xffb128);
info.addImages(assetVO.iconImage, assetVO.mediumImage, assetVO.largeImage);
} else
{
info.addTitle("!!" + dialog.getValue('speaker') + "!!", 0xffb128);
info.addImages('', '', '');
}
}

public function get currentMission():MissionVO { return _missionModel.currentMission; }
public function get fteStep():int { return _fteController.step; }
[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }
[Inject]
public function set game( v:Game ):void { _game = v; }
[Inject]
public function set missionModel( v:MissionModel ):void { _missionModel = v; }
[Inject]
public function set achievementModel( v:AchievementModel ):void { _achievementModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set sectorModel( v:SectorModel ):void { _sectorModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public
```

```
function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

override public function destroy():void
{
_assetModel = null;
_fleetModel = null;
_game = null;
_locToken = null;
_achievementModel = null;
_missionModel = null;
_prototypeModel = null;
_sectorModel = null;
_serverController = null;
_starbaseModel = null;
_transactionController = null;
}
}
}
```

File 405: igw\com\presenter\starbase\ShipyardPresenter.as

```
package com.presenter.starbase
{
import com.controller.ServerController;
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.TypeEnum;
import com.event.TransactionEvent;
import com.event.signal.TransactionSignal;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.ShipVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionVO;
import com.presenter.ImperiumPresenter;

public class ShipyardPresenter extends ImperiumPresenter implements IShipyardPresenter
{
private static var _tempID:int = 0;

private
```

```
var _assetModel:AssetModel;
private var _fleetModel:FleetModel;
private var _prototypeModel:PrototypeModel;
private var _starbaseModel:StarbaseModel;
private var _serverController:ServerController;
private var _transactionController:TransactionController;

/** Currently selected ship the player is working on, i.e. what will get built when they click "build"
*/
private var _currentShip:ShipVO = new ShipVO();

/** When a player refits a ship, this gets set. Used to continue refitting a ship when canceled */
private var _refittingShip:ShipVO;

/** Stores the ship design we have been working on this session so it doesn't get stomped by
scrapping, refitting etc. */
private static var _savedShip:ShipVO = new ShipVO();

[PostConstruct]
override public function init():void
{
super.init();
_transactionController.shipyardPresenter = this;
}

public function loadIcon( url:String, callback:Function ):void
{
_assetModel.getFromCache("assets/" + url, callback);
}

public function loadIconFromEntityData( type:String, callback:Function ):void
{
var _currentAssetVO:AssetVO = _assetModel.getEntityData(type);
loadIcon(_currentAssetVO.smallImage, callback);
}

public function buildShip( ship:ShipVO, purchaseType:uint ):void
{
ship.id = tempID;
_fleetModel.addShip(ship);
_transactionController.dockBuildShip(ship, purchaseType);
}

public function refitShip( ship:ShipVO, purchaseType:uint ):void
{
var existingShip:ShipVO = getShipByID(ship.id);
existingShip.refitShipName = ship.refitShipName;
existingShip.refitModules = ship.refitModules;
existingShip.calculateCosts();
_transactionController.dockRefitShip(existingShip, purchaseType);
}
```

```
public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getStatPrototypeByName( name:String ):IPrototype
{
var proto:IPrototype = _prototypeModel.getStatPrototypeByName(name);
return proto;
}

public function getModules( slotType:String ):Vector.<IPrototype> { return
_prototypeModel.getModulesBySlotType(slotType); }

public function getSlotPrototype( key:String ):IPrototype { return
_prototypeModel.getSlotPrototype(key); }

public function getBuildingShip():ShipVO
{
var ship:ShipVO;
var transaction:TransactionVO = _transactionController.getShipyardTransaction();
if (transaction)
ship = _fleetModel.getShip(transaction.id);
if (ship)
{
if (ship.refiting)
{
_refittingShip = ship.clone();
_refittingShip.id = ship.id;
_refittingShip.built = true;
} else
_refittingShip = null;
}
return ship;
}

public function getShipByID( id:String ):ShipVO { return _fleetModel.getShip(id); }

public function cancelTransaction( transaction:TransactionVO ):void
{
if (transaction)
{
_transactionController.transactionCancel(transaction.id);
}
}

public function recycleShip( shipVO:ShipVO ):void
{
```

```
if (shipVO)
{
_transactionController.dockRecycleShip(shipVO);
}
}

public function isResearched( tech:String ):Boolean
{
if (tech == '')
return true;

var requiredBuildingClass:String =
_prototypeModel.getResearchPrototypeByName(tech).getValue('requiredBuildingClass');
return _transactionController.isResearched(tech, requiredBuildingClass);
}

public function canBuild( ship:IPrototype ):RequirementVO { return
_transactionController.canBuildShip(ship); }

public function getAssetVO( prototype:IPrototype ):AssetVO
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;
return _assetModel.getEntityData(assetName);
}

public function isShipyardRepairing():Boolean
{
var repairing:Boolean = false;
var shipyard:BuildingVO =
_starbaseModel.getBuildingByClass(TypeEnum.CONSTRUCTION_BAY);
if (shipyard)
{
var transaction:TransactionVO =
_transactionController.getStarbaseBuildingTransaction(shipyard.constructionCategory,
shipyard.id);
if (transaction && transaction.type == TransactionEvent.STARBASE_REPAIR_BASE)
repairing = true;
}

return repairing;
}

public function getPrototypeByName( proto:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getBuildingPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getResearchPrototypeByName(proto);
if
```

```
(!iproto)
iproto = _prototypeModel.getShipPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getStoreItemPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(proto);
return iproto;
}

public function addTransactionListener( listener:Function ):void {
_transactionController.addListener(TransactionSignal.TRANSACTION, listener); }
public function removeTransactionListener( listener:Function ):void {
_transactionController.removeListener(listener); }

public function get currentShip():ShipVO { return _currentShip; }
public function set currentShip( value:ShipVO ):void { _currentShip = value; }

public function get builtShipCount():Number { return _fleetModel.builtShipCount; }
public function get maxAvailableShipSlots():Number { return
_fleetModel.maxAvailableShipSlots; }
public function get canBuildNewShips():Boolean { return _fleetModel.canBuildNewShips; }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }

[Inject]
public function set fleetModel( v:FleetModel ):void { _fleetModel = v; }

[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }

[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }

public function get refittingShip():ShipVO
{
if (_refittingShip)
{
//update the modules in case the ship has finished its refit
var existingShip:ShipVO = _fleetModel.getShip(_refittingShip.id);
if (existingShip)
_refittingShip.modules = existingShip.modules;
else
_refittingShip = null;
}
return _refittingShip;
}
public function get savedShip():ShipVO { return _savedShip; }
public function set savedShip( value:ShipVO ):void { _savedShip = value; }
[Inject]
public
```

```
function set serverController( v:ServerController ):void { _serverController = v; }
public function get shipPrototypes():Vector.<IPrototype> { return
_prototypeModel.getShipPrototypesByFaction(CurrentUser.faction); }
public function get shipyardTransaction():TransactionVO { return
_transactionController.getShipyardTransaction(); }
private function get tempID():String { _tempID++; return CurrentUser.name + '.clientside_ship.' +
_tempID; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

override public function destroy():void
{
_assetModel = null;
_fleetModel = null;
_prototypeModel = null;
_starbaseModel = null;
_serverController = null;
_transactionController.shipyardPresenter = null;
_transactionController = null;
}
}
}



-----------------------------------
File 406: igw\com\presenter\starbase\StarbasePresenter.as
package com.presenter.starbase
{
import com.controller.transaction.TransactionController;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.CurrencyEnum;
import com.enum.StarbaseConstructionEnum;
import com.enum.TypeEnum;
import com.enum.server.PurchaseTypeEnum;
import com.event.StarbaseEvent;
import com.event.TransactionEvent;
import com.event.signal.TransactionSignal;
import com.game.entity.factory.IStarbaseFactory;
import com.game.entity.systems.interact.StarbaseInteractSystem;
import com.game.entity.systems.starbase.StarbaseSystem;
import com.model.alliance.AllianceModel;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintModel;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BaseVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.StarbaseModel;
import
```

```
com.model.transaction.TransactionVO;
import com.presenter.shared.GamePresenter;

import org.ash.core.Entity;
import org.osflash.signals.Signal;
import org.parade.core.ViewController;

public class StarbasePresenter extends GamePresenter implements IStarbasePresenter
{
private var _instant:Boolean;
private var _onBaseInteraction:Signal;
private var _showedBuildings:Boolean;
private var _starbaseFactory:IStarbaseFactory;
private var _starbaseModel:StarbaseModel;
private var _blueprintModel:BlueprintModel;
private var _starbaseSystem:StarbaseSystem;
private var _system:StarbaseInteractSystem;
private var _transactionController:TransactionController;
private var _viewController:ViewController;
private var _allianceModel:AllianceModel;

[PostConstruct]
override public function init():void
{
super.init();
_instant = false;
_onBaseInteraction = new Signal(int, int, Entity);
_showedBuildings = false;
_starbaseSystem = StarbaseSystem(_game.getSystem(StarbaseSystem));
_system = StarbaseInteractSystem(_game.getSystem(StarbaseInteractSystem));
_system.presenter = this;
}

public function cancelTransaction( transaction:TransactionVO ):void {
_transactionController.transactionCancel(transaction.id); }
public function moveEntity():void { _system.setState(StarbaseInteractSystem.MOVE_STATE); }
public function onInteractionWithBaseEntity( x:int, y:int, baseEntity:Entity ):void
{
_onBaseInteraction.dispatch(x, y, baseEntity);
}

public function performTransaction( transactionType:String, prototype:IPrototype,
purchaseType:uint, ... args ):void
{
var buildingVO:BuildingVO;
switch (transactionType)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:

StarbaseInteractSystem(_game.getSystem(StarbaseInteractSystem)).buildFromPrototype(prototype,
purchaseType);
```

```
break;

case TransactionEvent.STARBASE_BUILDING_UPGRADE:
buildingVO = BuildingVO(prototype);
var upgradeVO:IPrototype =
_prototypeModel.getBuildingPrototype(buildingVO.getValue('upgrade'));
if (_transactionController.starbaseUpgradeBuilding(buildingVO, upgradeVO, purchaseType))
{
//if the upgrade is instant then we want to update the prototype
if (purchaseType == PurchaseTypeEnum.INSTANT)
{
buildingVO.prototype = upgradeVO;
}
}
break;

case TransactionEvent.STARBASE_BUILDING_RECYCLE:
//remove the building from the game
buildingVO = BuildingVO(prototype);
_starbaseModel.removeBuildingByID(buildingVO.id);

//deposit the recycle profits into the player's base
var baseVO:BaseVO = _starbaseModel.currentBase;
baseVO.deposit(Math.floor(buildingVO.alloyCost * 0.20), CurrencyEnum.ALLOY);
baseVO.deposit(Math.floor(buildingVO.creditsCost * 0.20), CurrencyEnum.CREDIT);
baseVO.deposit(Math.floor(buildingVO.energyCost * 0.20), CurrencyEnum.ENERGY);
baseVO.deposit(Math.floor(buildingVO.syntheticCost * 0.20), CurrencyEnum.SYNTHETIC);

if (buildingVO.itemClass == TypeEnum.PYLON)
_starbaseSystem.findPylonConnections(_game.getEntity(buildingVO.id), true);

//send to the server
_transactionController.starbaseRecycleBuilding(buildingVO);
_starbaseFactory.destroyStarbaseItem(_game.getEntity(buildingVO.id));
break;

case TransactionEvent.STARBASE_REFIT_BUILDING:
buildingVO = BuildingVO(prototype);
_transactionController.starbaseRefitBuilding(buildingVO, args[0], purchaseType);
if (purchaseType == PurchaseTypeEnum.INSTANT)
{
buildingVO.modules = args[0];
//update the turret graphic
/*if (buildingVO.asset == TypeEnum.POINT_DEFENSE_PLATFORM || buildingVO.asset ==
TypeEnum.SHIELD_GENERATOR)
{
_starbaseFactory.updateStarbaseBuilding(_game.getEntity(buildingVO.id));
}*/
} else
buildingVO.refitModules
```

```
= args[0];
break;

case TransactionEvent.STARBASE_RESEARCH:
_transactionController.starbaseStartResearch(prototype, purchaseType);
break;

case TransactionEvent.STARBASE_REPAIR_BASE:
_transactionController.starbaseRepairBuildings(getRepairCost(), purchaseType);
break;

default:
throw new Error("Unable to perform transaction of type " + transactionType);
break;
}
}

public function getFilterAssetVO( prototype:IPrototype ):AssetVO { return
_assetModel.getEntityData(prototype.getValue('filterCategory')); }
public function getBuildingUpgrade( buildingVO:BuildingVO ):IPrototype { return
_prototypeModel.getBuildingPrototype(buildingVO.getValue('upgrade')); }
public function getBuildingVO( id:String ):BuildingVO { return
_starbaseModel.getBuildingByID(id); }
public function getBuildingVOByClass( itemClass:String, highestLevel:Boolean = false
):BuildingVO { return _starbaseModel.getBuildingByClass(itemClass, highestLevel); }
public function getEntityName( assetName:String ):String { return
_assetModel.getEntityData(assetName).visibleName; }

public function getPrototypeByName( proto:String ):IPrototype
{
var iproto:IPrototype = _prototypeModel.getBuildingPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getResearchPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getShipPrototype(proto);
if (!iproto)
iproto = _prototypeModel.getStoreItemPrototypeByName(proto);
if (!iproto)
iproto = _prototypeModel.getWeaponPrototype(proto);
return iproto;
}

public function getBlueprintByName( prototype:String ):BlueprintVO
{
return _blueprintModel.getBlueprintByName(prototype);
}

public function getRequirements( transactionType:String, prototype:IPrototype ):RequirementVO
{
switch (transactionType)
{
```

```
case TransactionEvent.STARBASE_BUILDING_BUILD:
return _transactionController.canBuild(prototype);
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
return _transactionController.canUpgrade(BuildingVO(prototype));
case TransactionEvent.STARBASE_BLUEPRINT_PURCHASE:
case TransactionEvent.STARBASE_RESEARCH:
return _transactionController.canPurchaseResearch(prototype);
case TransactionEvent.STARBASE_REFIT_BUILDING:
return _transactionController.canRefit(prototype);

}
throw new Error("Unable to find requirements of transaction type " + transactionType);
return null;
}

public function getRepairCost():int
{
return _transactionController.getHardCurrencyCostFromSeconds(getRepairTime(true));
}

/**
* @param getTotal If true calculates the total time of all buildings otherwise calculates the
longest of one building
* @return The total time to repair a base or the longest amount of time to repair one building
*/
public function getRepairTime( getTotal:Boolean = false ):int
{
var building:BuildingVO;
var localTime:int = 0;
var time:int = 0;
for (var i:int = 0; i < _starbaseModel.buildings.length; i++)
{
building = _starbaseModel.buildings[i];
if (building.currentHealth == 1)
continue;
localTime = building.prototype.getValue("repairTimeSeconds");
for each (var proto:IPrototype in building.modules)
{
if (!proto)
continue;
localTime += proto.getValue("repairTimeSeconds");
}
if (getTotal)
time += localTime * (1 - building.currentHealth);
else
{
localTime = localTime * (1 - building.currentHealth);
if (localTime > time)
time = localTime;
}
```

```
}
return time;
}

public function getSlotType( key:String ):String { return
_prototypeModel.getSlotPrototype(key).getValue('slotType'); }
public function getStarbaseBuildingTransaction( constructionCategory:String = null,
buildingID:String = null ):TransactionVO { return
_transactionController.getStarbaseBuildingTransaction(constructionCategory, buildingID); }
public function getStarbaseResearchTransaction( buildingType:String ):TransactionVO { return
_transactionController.getStarbaseResearchTransactionByBuildingType(buildingType); }
public function loadIcon( url:String, callback:Function ):void {
_assetModel.getFromCache("assets/" + url, callback); }

public function showBuildings():void
{
if (!_showedBuildings)
{
StarbaseSystem(_game.getSystem(StarbaseSystem)).createBuildingsFromStarbase();
_showedBuildings = true;
}
}

public function getBlueprintHardCurrencyCost( blueprint:BlueprintVO, partsPurchased:Number
):Number
{
return _transactionController.getBlueprintHardCurrencyCost(blueprint, partsPurchased);
}

public function purchaseBlueprint( blueprint:BlueprintVO, partsPurchased:Number ):void
{
_transactionController.buyBlueprintTransaction(blueprint, partsPurchased);
}
public function completeBlueprintResearch( blueprint:BlueprintVO):void
{
_transactionController.completeBlueprintResearchTransaction(blueprint);
}

public function addBaseInteractionListener( callback:Function ):void {
_onBaseInteraction.add(callback); }

public function addTransactionListener( listener:Function ):void {
_transactionController.addListener(TransactionSignal.TRANSACTION, listener); }
public function removeTransactionListener( listener:Function ):void {
_transactionController.removeListener(listener); }

public function addOnGenericAllianceMessageRecievedListener( callback:Function ):void {
_allianceModel.onGenericAllianceMessageRecieved.add(callback); }
public function removeOnGenericAllianceMessageRecievedListener( callback:Function ):void {
_allianceModel.onGenericAllianceMessageRecieved.remove(callback);
```

```
}

public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getStatPrototypeByName( name:String ):IPrototype
{
var proto:IPrototype = _prototypeModel.getStatPrototypeByName(name);
return proto;
}

override public function confirmReady():void
{
super.confirmReady();
dispatch(new StarbaseEvent(StarbaseEvent.WELCOME_BACK));
}

public function get buildingPrototypes():Vector.<IPrototype> { return
_prototypeModel.getBuildableBuildingPrototypes(); }
public function get currentBase():BaseVO { return _starbaseModel.currentBase; }
public function get researchPrototypes():Vector.<IPrototype> { return
_prototypeModel.getResearchPrototypes(); }

public function get totalDamagedBuildings():int
{
var damaged:int = 0;
var buildings:Vector.<BuildingVO> = _starbaseModel.buildings;
if (buildings)
{
for (var i:int = 0; i < buildings.length; i++)
{
if (buildings[i].currentHealth < 1 && buildings[i].currentHealth > 0)
damaged++;
}
}
return damaged;
}

public function get totalDestroyedBuildings():int
{
var destroyed:int = 0;
var buildings:Vector.<BuildingVO> = _starbaseModel.buildings;
if (buildings)
{
for (var i:int = 0; i < buildings.length; i++)
{
if (buildings[i].currentHealth == 0)
destroyed++;
```

```
}
}
return destroyed;
}

public function get totalBaseDamage():Number
{
var health:Number = 0;
var buildings:Vector.<BuildingVO> = _starbaseModel.buildings;
var maxHealth:Number = 0;
if (buildings)
{
for (var i:int = 0; i < buildings.length; i++)
{
if (buildings[i].constructionCategory != StarbaseConstructionEnum.PLATFORM)
{
health += buildings[i].currentHealth * 100;
maxHealth += 100;
}
}
}
return Math.round((1 - (health / maxHealth)) * 100);
}

[Inject]
public function set starbaseFactory( v:IStarbaseFactory ):void { _starbaseFactory = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set blueprintModel( v:BlueprintModel ):void { _blueprintModel = v; }
[Inject]
public function set allianceModel( v:AllianceModel ):void { _allianceModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }
[Inject]
public function set viewController( v:ViewController ):void { _viewController = v; }

override public function destroy():void
{
super.destroy();
_onBaseInteraction.removeAll();
_onBaseInteraction = null;
_starbaseFactory = null;
_starbaseModel = null;
_blueprintModel = null;
_system = null;
_transactionController = null;
_viewController = null;
}
```

```
}
}

-----------------------------------
File 407: igw\com\presenter\starbase\StorePresenter.as
package com.presenter.starbase
{
import com.Application;
import com.controller.transaction.TransactionController;
import com.enum.server.PurchaseTypeEnum;
import com.event.signal.TransactionSignal;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.fleet.FleetModel;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.starbase.StarbaseModel;
import com.model.transaction.TransactionModel;
import com.presenter.ImperiumPresenter;
import com.service.server.incoming.data.BuffData;

import flash.utils.Dictionary;
import flash.utils.getTimer;

import org.shared.ObjectPool;

public class StorePresenter extends ImperiumPresenter implements IStorePresenter
{
[Inject]
public var assetModel:AssetModel;
[Inject]
public var prototypeModel:PrototypeModel;
[Inject]
public var starbaseModel:StarbaseModel;
[Inject]
public var fleetModel:FleetModel;
[Inject]
public var transactionController:TransactionController;
[Inject]
public var transactionModel:TransactionModel;


private static var _tempID:int = 0;
private static var _tempOtherID:int = 0;

public
```

```
function getCurrentState():String
{
return Application.STATE;
}

public function getStoreItemPrototypes():Vector.<IPrototype>
{
return prototypeModel.getStoreItemPrototypes();
}

public function getPrototypeUIName( prototype:IPrototype ):String
{
var assetName:String = prototype.uiAsset;
if (!assetName)
assetName = prototype.asset;

var currentAssetVO:AssetVO = assetModel.getEntityData(assetName);
return currentAssetVO.visibleName;
}

public function getProtoTypeUIDescriptionText( prototype:IPrototype ):String
{
var currentAssetVO:AssetVO = assetModel.getEntityData(prototype.getValue('uiAsset'));
return currentAssetVO.descriptionText;
}

public function getPrototypeUISmallImage( prototype:IPrototype ):String
{
var currentAssetVO:AssetVO = assetModel.getEntityData(prototype.getValue('uiAsset'));
return currentAssetVO.smallImage;
}

public function loadIconFromPrototype( prototype:IPrototype, callback:Function ):void
{
assetModel.getFromCache("assets/" + getPrototypeUISmallImage(prototype), callback);
}

public function speedUpTransaction( serverKey:String, token:int, instant:Boolean,
speedUpBy:int, fromStore:Boolean, cost:int ):void
{
transactionController.speedUpTransaction(serverKey, token, instant, speedUpBy, fromStore,
cost);
}

public function buyResourceTransaction( prototype:IPrototype, percent:int, centerBase:Boolean,
cost:int ):void
{
transactionController.buyResourceTransaction(prototype, percent, centerBase, cost);
}

public
```

```
function buyItemTransaction( buffPrototype:IPrototype, centerBase:Boolean, cost:int ):void
{
//save a new buff in the same way that the server would send it to us
var buffData:BuffData = ObjectPool.get(BuffData);
buffData.baseID = starbaseModel.currentBaseID;
buffData.id = tempID;
buffData.playerOwnerID = CurrentUser.id;
buffData.prototype = prototypeModel.getBuffPrototype(buffPrototype.name);
buffData.began = getTimer();
buffData.ends = buffData.began + buffData.prototype.getValue("durationSeconds") * 1000;
buffData.now = buffData.began;
starbaseModel.importBuffData(buffData);
transactionController.buyStoreItemTransaction(buffPrototype, centerBase, buffData.id, cost);
}

public function buyOtherItemTransaction( prototype:IPrototype, amount:int, centerBase:Boolean,
cost:int ):void
{
transactionController.buyOtherStoreItemTransaction(prototype, amount, centerBase,
tempOtherID, cost);
}

public function getHardCurrencyCostFromSeconds( buildTimeSeconds:Number ):int
{
return transactionController.getHardCurrencyCostFromSeconds(buildTimeSeconds);
}

public function getHardCurrencyCostFromResource( resources:int, type:String ):int
{
return transactionController.getHardCurrencyCostFromResource(resources, type);
}

public function getCanAfford( prototype:IPrototype ):Boolean
{
return transactionController.checkCost(prototype, PurchaseTypeEnum.NORMAL, true);
}

public function getMaxResources():uint
{
return starbaseModel.maxResources
}

public function getMaxCredits():uint
{
return starbaseModel.maxCredits;
}

public function getResourceCount( type:String ):uint
{
return starbaseModel.getCurrentResourceCount(type);
}
```

```
public function getShipById( id:String ):ShipVO
{
return fleetModel.getShip(id);
}

public function getFleetById( id:String ):FleetVO
{
return fleetModel.getFleet(id);
}

public function getBuildingByID( id:String ):BuildingVO
{
return starbaseModel.getBuildingByID(id);
}

public function getResearchByID( id:String ):ResearchVO
{
return starbaseModel.getResearchByID(id);
}

public function getBuildingVOByClass( itemClass:String ):BuildingVO
{
return starbaseModel.getBuildingByClass(itemClass, false);
}

public function addOnTransactionUpdatedListener( callback:Function ):void {
transactionModel.addListener(TransactionSignal.TRANSACTION_UPDATED, callback); }
public function removeOnTransactionUpdatedListener( callback:Function ):void {
transactionModel.removeListener(callback); }

public function addOnTransactionRemovedListener( callback:Function ):void {
transactionModel.addListener(TransactionSignal.TRANSACTION_REMOVED, callback); }
public function removeOnTransactionRemovedListener( callback:Function ):void {
transactionModel.removeListener(callback); }

public function get tempID():String { _tempID++; return 'clientSide.buff' + _tempID; }
public function get tempOtherID():String { _tempOtherID++; return 'clientSide.otherItem' +
_tempOtherID; }

public function getTransactions():Dictionary
{
return transactionModel.transactions;
}
}
}
```

File 408: igw\com\presenter\starbase\TradePresenter.as

```
package com.presenter.starbase
{
import com.controller.SettingsController;
import com.controller.transaction.TransactionController;
import com.event.signal.TransactionSignal;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.StarbaseModel;
import com.model.starbase.TradeRouteVO;
import com.model.transaction.TransactionVO;
import com.presenter.ImperiumPresenter;
import com.util.TradeRouteUtil;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class TradePresenter extends ImperiumPresenter implements ITradePresenter
{
protected const _logger:ILogger = getLogger('TradePresenter');

private var _assetModel:AssetModel;
private var _prototypeModel:PrototypeModel;
private var _settingsController:SettingsController;
private var _starbaseModel:StarbaseModel;
private var _transactionController:TransactionController;

[PostConstruct]
override public function init():void
{
super.init();
}

public function getConstantPrototypeValueByName( name:String ):Number
{
var proto:IPrototype = _prototypeModel.getConstantPrototypeByName(name);
return proto.getValue('value');
}

public function getProtoTypeUIName( prototype:IPrototype ):String
{
var currentAssetVO:AssetVO = _assetModel.getEntityData(prototype.getValue('uiAsset'));
if (currentAssetVO != null)
return currentAssetVO.visibleName;
else
return '';
}

public
```

```
function getPrototypeUIDescription( prototype:IPrototype ):String
{
var currentAssetVO:AssetVO = _assetModel.getEntityData(prototype.getValue('uiAsset'));
if (currentAssetVO != null)
return currentAssetVO.descriptionText;
else
return '';
}

public function loadIconFromPrototype( type:String, prototype:IPrototype, callback:Function
):void
{

var currentAssetVO:AssetVO = _assetModel.getEntityData(prototype.getValue('uiAsset'));
if (currentAssetVO != null)
_assetModel.getFromCache('assets/' + currentAssetVO[type], callback);
}

public function getTradeRouteTransaction( id:String ):TransactionVO
{
return _transactionController.getTradeRouteTransaction(id);
}

public function getContractsFromFaction( contractGroup:String ):Vector.<IPrototype>
{
var contracts:Vector.<IPrototype> = new Vector.<IPrototype>;
var allContracts:Vector.<IPrototype> = _prototypeModel.getContractPrototypes();
var len:uint = allContracts.length;
var currentPrototype:IPrototype;
for (var i:uint = 0; i < len; ++i)
{
currentPrototype = allContracts[i];
if (contractGroup == currentPrototype.getValue('contractGroup'))
contracts.push(currentPrototype);
}

return contracts;
}

public function getBalancedContractFromFaction( contractGroup:String ):IPrototype
{
var balancedContract:IPrototype;
var typeOfContract:String;
var allContracts:Vector.<IPrototype> = _prototypeModel.getContractPrototypes();
var len:uint = allContracts.length;
var currentPrototype:IPrototype;

switch (contractGroup)
{
case 'Starwind':
typeOfContract
```

```
= 'SyntheticsPreferred';
break;
case 'Solaris':
typeOfContract = 'EnergyPreferred';
break;
case 'BlueComet':
typeOfContract = 'CreditsPreferred';
break;
case 'WhiteDwarf':
typeOfContract = 'AlloyPreferred';
break;
case 'AcmeUniversal':
typeOfContract = 'Balanced';
break;

}


for (var i:uint = 0; i < len; ++i)
{
currentPrototype = allContracts[i];
if (contractGroup == currentPrototype.getValue('contractGroup') &&
String(currentPrototype.name).indexOf(typeOfContract) != -1)
balancedContract = currentPrototype;
}

return balancedContract;
}

public function getAgentsFromFaction( contractGroup:String ):Vector.<IPrototype>
{
var agents:Vector.<IPrototype> = new Vector.<IPrototype>;
var allAgents:Vector.<IPrototype> = _prototypeModel.getAgentPrototypes();
var len:uint = allAgents.length;
var currentPrototype:IPrototype;
for (var i:uint = 0; i < len; ++i)
{
currentPrototype = allAgents[i];
if (contractGroup == currentPrototype.getValue('faction'))
agents.push(currentPrototype);
}

return agents;
}

public function getAgent( contractGroup:String, reputation:Number ):IPrototype
{
var agents:Vector.<IPrototype> = getAgentsFromFaction(contractGroup);
var len:uint = agents.length;
var currentAgent:IPrototype;
for
```

```
(var i:uint = 0; i < len; ++i)
{
currentAgent = agents[i];
if (currentAgent.getValue('minRep') <= reputation && currentAgent.getValue('maxRep') >=
reputation)
break;
}

return currentAgent;
}

public function hasAgentGreetingBeenViewed( agentID:int ):Boolean
{
return _settingsController.hasAgentGreetingBeenViewed(agentID);
}

public function setAgentGreetingViewed( agentID:int ):void
{
_settingsController.setAgentGreetingViewed(agentID);
}

public function getAgentDialogByGroup( group:String ):Vector.<IPrototype>
{
var dialog:Vector.<IPrototype> = new Vector.<IPrototype>;
var allDialog:Vector.<IPrototype> = _prototypeModel.getDialogPrototypes();
var len:uint = allDialog.length;
var currentPrototype:IPrototype;
for (var i:uint = 0; i < len; ++i)
{
currentPrototype = allDialog[i];
if (group == currentPrototype.getValue('dialogGroup'))
dialog.push(currentPrototype);
}

return dialog;
}

public function requestContract( centerSpaceBase:Boolean, contractPrototype:String,
factionPrototype:String, callback:Function ):void
{
var security:Number = getConstantPrototypeValueByName('contractSecurityDefault');
var duration:Number = getConstantPrototypeValueByName('contractDurationDefault');
var frequency:Number = getConstantPrototypeValueByName('contractFrequencyDefault');
var payout:Number = getConstantPrototypeValueByName('contractPayoutDefault');
var productivity:Number = getConstantPrototypeValueByName('contractProductivityDefault');

_transactionController.requestContract(centerSpaceBase, contractPrototype, duration,
factionPrototype, frequency, payout, productivity, security, callback, true);
}

public
```

```
function cancelContract( id:String ):void
{
_transactionController.transactionCancel(id);
}

public function addTransactionListener( callback:Function ):void {
_transactionController.addListener(TransactionSignal.TRANSACTION, callback); }
public function removeTransactionListener( callback:Function ):void {
_transactionController.removeListener(callback); }

public function get tradeRouteCreditIncome():uint { return
_starbaseModel.tradeRouteCreditIncome; }
public function get tradeRouteResourceIncome():uint { return
_starbaseModel.tradeRouteResourceIncome; }

public function get maxContracts():int { return TradeRouteUtil.maxContracts; }
public function get maxUnlockedContracts():int { return TradeRouteUtil.maxUnlockedContracts;
}

public function get tradeRoutes():Vector.<TradeRouteVO> { return
_starbaseModel.getTradeRoutesByBaseID(); }

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set prototypeModel( v:PrototypeModel ):void { _prototypeModel = v; }
[Inject]
public function set settingsController( v:SettingsController ):void { _settingsController = v; }
[Inject]
public function set starbaseModel( v:StarbaseModel ):void { _starbaseModel = v; }
[Inject]
public function set transactionController( v:TransactionController ):void { _transactionController
= v; }

override public function destroy():void
{
super.destroy();
}
}
}
```

File 409: igw\com\service\ExternalInterfaceAPI.as

```
package com.service
{
import com.Application;
import com.controller.SettingsController;
import com.event.TransactionEvent;
import
```

```
com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.service.kongregate.KongregateAPI;
import com.service.language.Localization;

import flash.display.StageDisplayState;
import flash.external.ExternalInterface;
import flash.net.URLRequest;
import flash.net.navigateToURL;
import flash.utils.Dictionary;
import flash.utils.getQualifiedClassName;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;



public class ExternalInterfaceAPI
{

static private const ON_SWF_LOADED:String = "Imperium.onSwfLoaded";

static private const SHARE_PvP_VICTORY:String = "Imperium.sharePvPVictory";
static private const SHARE_USER_LEVEL_UP:String = "Imperium.shareLevelUp";
static private const SHARE_TRANSACTION:String = "Imperium.shareTransaction";
static private const SHARE_PLAY_REQUEST:String = "Imperium.sharePlayRequest";

static private const GET_FACEBOOK_USER_ID:String = "Imperium.getFacebookUserId";
static private const GET_FACEBOOK_USER_ACCESS_TOKEN:String =
"Imperium.getFacebookAccessToken";
static private const GET_FACEBOOK_ITEMS:String = "Imperium.getFacebookItems";
static private const OPEN_FACEBOOK_PAYWALL:String = "Imperium.openFacebookPaywall";

static private const GET_LOGIN_PROTOCOL:String = "Imperium.getLoginProtocol";
static private const GET_LOGIN_HOSTNAME:String = "Imperium.getLoginHostname";
static private const GET_LOGIN_PORT:String = "Imperium.getLoginPort";
static private const GET_PAYMENT_PROTOCOL:String = "Imperium.getPaymentProtocol";
static private const GET_PAYMENT_HOSTNAME:String = "Imperium.getPaymentHostname";
static private const GET_PAYMENT_PORT:String = "Imperium.getPaymentPort";
static private const OPEN_XSOLLA_STORE:String = "Imperium.openXsollaStore";
static private const SET_PLAY_PLATFORM:String = "Imperium.setPlayPlatform";
static private const RELOAD_SWF:String = "Imperium.reloadSwf";

static private const GET_BATTLE_WEB_PATH:String = "Imperium.getBattleWebPath";
static private const ON_REFRESH:String = "Imperium.onRefresh";
static private const ON_LOG_OUT:String = "Imperium.onLogOut";
static private const ON_REGISTER_GUEST:String = "Imperium.onRegisterGuest";

static
```

```
private const POP_PAYWALL:String = "Imperium.popPaywall";
static private const POP_PIXEL:String = "Imperium.popPixel";

static private const GET_LOCALIZATION_FOLDER:String = "Imperium.getLocalizationFolder";
static private const GET_LOCALIZATION_VO_FOLDER:String =
"Imperium.getLocalizationVOFolder";
static private const GET_MAIN_FONT:String = "Imperium.getMainFont";
static private const GET_TRACE_FONT:String = "Imperium.getTraceFont";
static private const GET_FONT:String = "Imperium.getFont";
static private const SET_LOCALIZATION_TAG:String = "Imperium.saveLocalizationTag";
static private const GET_LOCALIZATION_TAG:String = "Imperium.getLocalizationTag";

static private const GET_LANGUAGE_CODE:String = "Imperium.getLanguageCode";
static private const GET_COUNTRY_CODE:String = "Imperium.getCountryCode";
static private const GET_ENTRY_TAG:String = "Imperium.getEntryTag";

static private const PURCHASE_KONGREGATE_ITEMS:String =
"ImperiumKongregateDriver.purchaseKongregateItems";
static private const GET_PLAYER_KONGREGATE_USER_ID:String =
"ImperiumKongregateDriver.getPlayerKongregateUserId";
static private const GET_PLAYER_KONGREGATE_USERNAME:String =
"ImperiumKongregateDriver.getPlayerKongregateUsername";
static private const GET_PLAYER_KONGREGATE_GAME_AUTH_TOKEN:String =
"ImperiumKongregateDriver.getPlayerKongregateGameAuthToken";
static private const SUBMIT_KONGREGATE_STAT:String =
"ImperiumKongregateDriver.submitKongregateStat";

static private const GET_PLAYER_XSOLLA_GAME_AUTH_TOKEN:String =
"Imperium.getPlayerXsollaGameAuthToken";

static private const GET_STEAM_SESSION_TICKET:String =
"Imperium.getSteamSessionTicket";

static private const GET_GUEST_USER_ID:String = "Imperium.getGuestUserId";
static private const SET_GUEST_USER_ID:String = "Imperium.setGuestUserId";

static private const GET_GUEST_TO_XSOLLA_USER_ID:String =
"Imperium.getGuestToXsollaUserId";
static private const COMPLETE_GUEST_TO_XSOLLA:String =
"Imperium.completeGuestToXsollaUserId";

static private var FBSharePvP_name:String = "CodeString.FacebookSharePvP.Name";
static private var FBSharePvP_description:String =
"CodeString.FacebookSharePvP.Description";
static private var FBSharePvP_caption:String = "CodeString.FacebookSharePvP.Caption";

static private var FBSharePvB_name:String = "CodeString.FacebookSharePvB.Name";
static private var FBSharePvB_description:String =
"CodeString.FacebookSharePvB.Description";
static private var FBSharePvB_caption:String = "CodeString.FacebookSharePvB.Caption";

static
```

```
private var FBSharePlayRequest_description:String =
"CodeString.FacebookSharePlayRequest.Message";
static private var FBSharePlayRequest_title:String =
"CodeString.FacebookSharePlayRequest.Title";

private static var _instance:ExternalInterfaceAPI;

private var _assetModel:AssetModel;
private var _settingsController:SettingsController;
//private var _kongregateAPI:KongregateAPI;

[PostConstruct]
public function init():void
{
_instance = this;

if (ExternalInterface.available)
{
ExternalInterface.addCallback("setBatteryStatus", Application.setBatteryStatus);
ExternalInterface.addCallback("sharePlayRequest", sharePlayRequest);
//ExternalInterface.addCallback("onKongregateItemsPurchased",
_kongregateAPI.onKongregateItemsPurchased);

onSwfLoaded();
ExternalInterfaceAPI.logConsole("Imperium init ExternalAPI - success");
}
else
{
ExternalInterfaceAPI.logConsole("Imperium init ExternalAPI - failed");
}
}


public static function logConsole(msg:String, caller:Object = null):void{
var str:String = "";
if(caller){
str = getQualifiedClassName(caller);
str += ":: ";
}
str += msg;
trace(str);
if(ExternalInterface.available){
ExternalInterface.call("console.log", str);
}
}


private function toggleFullscreen():void
{
if (Application.STAGE.displayState == StageDisplayState.FULL_SCREEN_INTERACTIVE)
_settingsController.toggleFullScreen();
```

```
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set settingsController( v:SettingsController ):void { _settingsController = v; }
//[Inject]
//public function set kongregateAPI( v:KongregateAPI ):void { _kongregateAPI = v; }

public static function onSwfLoaded():void
{
if (ExternalInterface.available)
ExternalInterface.call(ON_SWF_LOADED);
}
public static function getFacebookUserId():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_FACEBOOK_USER_ID);
}
public static function getFacebookUserAccessToken():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_FACEBOOK_USER_ACCESS_TOKEN);
}
public static function GetFacebookItems():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_FACEBOOK_ITEMS);
}
public static function OpenFacebookPaywall(quantity:int):void
{
if (!ExternalInterface.available)
return;
ExternalInterface.call(OPEN_FACEBOOK_PAYWALL, quantity);
}

public static function getLoginProtocol():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_LOGIN_PROTOCOL);
}

public static function getLoginHostname():String
{
if (!ExternalInterface.available)
return
```

```
null;
return ExternalInterface.call(GET_LOGIN_HOSTNAME);
}

public static function getLoginPort():int
{
if (!ExternalInterface.available)
return 0;
return ExternalInterface.call(GET_LOGIN_PORT);
}

public static function getPaymentProtocol():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_PAYMENT_PROTOCOL);
}

public static function getPaymentHostname():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_PAYMENT_HOSTNAME);
}

public static function getPaymentPort():int
{
if (!ExternalInterface.available)
return 0;
return ExternalInterface.call(GET_PAYMENT_PORT);
}

public static function openXsollaStore(token:String):int
{
if(CONFIG::IS_DESKTOP){
var urlRequest:URLRequest = new
URLRequest("https://secure.xsolla.com/paystation2/?access_token=" + token);
navigateToURL(urlRequest);
return 1;
}
if (!ExternalInterface.available)
return 0;
return ExternalInterface.call(OPEN_XSOLLA_STORE, token);
}

public static function setPlayPlatform( num:int ):void
{
if (ExternalInterface.available)
ExternalInterface.call(SET_PLAY_PLATFORM, num);
}

public
```

```
static function reloadSWF():void
{
if(CONFIG::IS_DESKTOP){
var NativeApplicationTypeDef:Object = getDefinitionByName("flash.desktop.NativeApplication");
NativeApplicationTypeDef.nativeApplication.exit(0);
}

if (ExternalInterface.available)
ExternalInterface.call(RELOAD_SWF);
}

public static function getBattleWebPath():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_BATTLE_WEB_PATH);
}

public static function refresh():void
{
if(CONFIG::IS_DESKTOP){
var NativeApplicationTypeDef:Object = getDefinitionByName("flash.desktop.NativeApplication");
NativeApplicationTypeDef.nativeApplication.exit(0);
}

if (ExternalInterface.available)
ExternalInterface.call(ON_REFRESH);
}

public static function registerGuest():void
{
if (ExternalInterface.available)
ExternalInterface.call(ON_REGISTER_GUEST);
}

public static function logOut():void
{
if (ExternalInterface.available)
ExternalInterface.call(ON_LOG_OUT);
}

public static function popPixel( num:int ):void
{
if (ExternalInterface.available)
ExternalInterface.call(POP_PIXEL, num);
}

public static function popPayWall():void
{
_instance.toggleFullscreen();

if
```

```
(ExternalInterface.available)
{
ExternalInterface.call(POP_PAYWALL, CurrentUser.naid, CurrentUser.oAuthID,
CurrentUser.language);
}
}

public static function getLocalizationFolder():String
{
if (!ExternalInterface.available)
return "001_en";
return ExternalInterface.call(GET_LOCALIZATION_FOLDER);
}

public static function getLocalizationVOFolder():String
{
if (!ExternalInterface.available)
return "001_en";
return ExternalInterface.call(GET_LOCALIZATION_VO_FOLDER);
}

public static function getMainFont():String
{
if (!ExternalInterface.available)
return "Arial";
return ExternalInterface.call(GET_MAIN_FONT);
}
public static function getTraceFont():String
{
if (!ExternalInterface.available)
return "Verdana";
return ExternalInterface.call(GET_TRACE_FONT);
}
public static function getFont(nr:int):String
{
if (!ExternalInterface.available)
return (nr == 1) ? "Open Sans" : "Agency FB";
return ExternalInterface.call(GET_FONT, nr);
}

public static function setLocalizationTag(tag:String):void
{
if (ExternalInterface.available)
{
ExternalInterface.call(SET_LOCALIZATION_TAG, tag);
}
}
public static function getLocalizationTag():String
{
if (!ExternalInterface.available)
return
```

```
"en";
return ExternalInterface.call(GET_LOCALIZATION_TAG);
}
public static function getLanguageCode():String
{
if (!ExternalInterface.available)
return "en";
if(CONFIG::IS_DESKTOP){
//return Imperium.language;
return "en";
}
return ExternalInterface.call(GET_LANGUAGE_CODE);
}
public static function getCountryCode():String
{
if (!ExternalInterface.available)
return "US";
if(CONFIG::IS_DESKTOP){
//return Imperium.country;
return "US";
}
return ExternalInterface.call(GET_COUNTRY_CODE);
}
public static function getEntryTag():String
{
if (!ExternalInterface.available)
return "";
if(CONFIG::IS_DESKTOP){
//return Imperium.entryTag;
return "";
}
return ExternalInterface.call(GET_ENTRY_TAG);
}

public static function submitKongregateStat( name:String, value:int ):void
{
if (ExternalInterface.available)
{
ExternalInterface.call(SUBMIT_KONGREGATE_STAT, name, value);
}
}

public static function purchaseKongregateItems( items:Array ):void
{
if (ExternalInterface.available)
{
ExternalInterface.call(PURCHASE_KONGREGATE_ITEMS, items);
}
}

public
```

```
static function getPlayerKongregateUserId():Number
{
if (!ExternalInterface.available)
return 0;
return ExternalInterface.call(GET_PLAYER_KONGREGATE_USER_ID);
}

public static function getPlayerKongregateUsername():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_PLAYER_KONGREGATE_USERNAME);
}

public static function getPlayerKongregateGameAuthToken():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_PLAYER_KONGREGATE_GAME_AUTH_TOKEN);
}

public static function getPlayerXsollaGameAuthToken():String
{
if(CONFIG::IS_DESKTOP){
return Imperium.authToken;
}

if (!ExternalInterface.available)
return null;

return ExternalInterface.call(GET_PLAYER_XSOLLA_GAME_AUTH_TOKEN);
}

public static function getSteamSessionTicket():String
{
if (!ExternalInterface.available)
return null;
return ExternalInterface.call(GET_STEAM_SESSION_TICKET);
}

public static function getGuestUserId():String
{
if (!ExternalInterface.available)
return null;

return ExternalInterface.call(GET_GUEST_USER_ID);
}

public static function getGuestToXsollaUserId():String
{
if
```

```
(!ExternalInterface.available)
return null;

return ExternalInterface.call(GET_GUEST_TO_XSOLLA_USER_ID);
}

public static function completeGuestToXsolla():void
{
if (!ExternalInterface.available)
return;

ExternalInterface.call(COMPLETE_GUEST_TO_XSOLLA);
}

public static function setGuestUserId(id:String):void
{
if (!ExternalInterface.available)
return;

ExternalInterface.call(SET_GUEST_USER_ID, id);
}

public static function sharePlayRequest():void
{
if (ExternalInterface.available)
{
_instance.toggleFullscreen();
var localizationObj:Dictionary = new Dictionary();
localizationObj["[[String.PlayerFaction]]"] = CurrentUser.faction;

if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
var og:OpenGraphShareObject = new OpenGraphShareObject();
og.description = Localization.instance.getStringWithTokens(FBSharePlayRequest_description,
localizationObj);
og.title = Localization.instance.getString(FBSharePlayRequest_title);

ExternalInterface.call(SHARE_PLAY_REQUEST + "_facebook", og);
}
}
}

public static function shareVictory( enemyPlayer:PlayerVO, isBaseCombat:Boolean ):void
{
if (ExternalInterface.available)
{
_instance.toggleFullscreen();
var localizationObj:Dictionary = new Dictionary();
localizationObj["[[String.PlayerName]]"] = CurrentUser.name;
localizationObj["[[String.PlayerFaction]]"] = CurrentUser.faction;
localizationObj["[[String.EnemyName]]"]
```

```
= enemyPlayer.name;
localizationObj["[[String.EnemyFaction]]"] = enemyPlayer.faction;

if (Application.NETWORK == Application.NETWORK_FACEBOOK || true)
{
var assetPrefix:String;

if (Application.ASSET_PATH != "" && Application.ASSET_PATH.indexOf('localhost') == -1)
assetPrefix = Application.ASSET_PATH + "assets/";
else
assetPrefix = "http://oddalchemy.com/kabam/";

var og:OpenGraphShareObject = new OpenGraphShareObject();
og.type = OpenGraphShareObject.ACTION_DEFEAT;
og.title = enemyPlayer.name;

if (isBaseCombat)
{
og.name = Localization.instance.getStringWithTokens(FBSharePvB_name, localizationObj);
og.description = Localization.instance.getStringWithTokens(FBSharePvB_description,
localizationObj);
og.caption = Localization.instance.getString(FBSharePvB_caption);
og.image = assetPrefix + getSocialImage("BaseCombat");
} else
{
og.name = Localization.instance.getStringWithTokens(FBSharePvP_name, localizationObj);
og.description = Localization.instance.getStringWithTokens(FBSharePvP_description,
localizationObj);
og.caption = Localization.instance.getString(FBSharePvP_caption);
og.image = assetPrefix + getSocialImage("PVPCombat");
}

ExternalInterface.call(SHARE_PvP_VICTORY + "_facebook", og);
}
}
}

public static function shareTransaction( type:String, prototype:IPrototype ):void
{
if (ExternalInterface.available)
{
if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
var asset:AssetVO = AssetModel.instance.getEntityData(prototype.uiAsset);
if (asset)
{
var og:OpenGraphShareObject = new OpenGraphShareObject();
og.type = resolveTransactionType_facebook(type);
og.title = resolveTransactionTitle(type, prototype, asset);
if (Application.ASSET_PATH != "" && Application.ASSET_PATH.indexOf('localhost') == -1)
og.image
```

```
= Application.ASSET_PATH + "assets/" + getSocialImage(type);
else
og.image = "http://oddalchemy.com/kabam/" + getSocialImage(type);
og.description = Localization.instance.getString(asset.descriptionText);
ExternalInterface.call(SHARE_TRANSACTION + "_facebook", og);
}
}
}
}

static private function resolveTransactionType_facebook( input:String ):String
{
var out:String;

switch (input)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILD_SHIP:
{
out = OpenGraphShareObject.ACTION_BUILD;
break;
}

case TransactionEvent.STARBASE_REFIT_BUILDING:
case TransactionEvent.STARBASE_REFIT_SHIP:
{
out = OpenGraphShareObject.ACTION_REFIT;
break;
}

case TransactionEvent.STARBASE_RESEARCH:
{
out = OpenGraphShareObject.ACTION_RESEARCH;
break;
}

case TransactionEvent.STARBASE_BUILDING_UPGRADE:
default:
{
out = OpenGraphShareObject.ACTION_UPGRADE;
break;
}
}

return out;
}

static private function resolveTransactionTitle( type:String, prototype:IPrototype, asset:AssetVO
):String
{
var
```

```
out:String;
var localizationObj:Dictionary = new Dictionary();
switch (type)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILD_SHIP:
case TransactionEvent.STARBASE_REFIT_SHIP:
case TransactionEvent.STARBASE_REFIT_BUILDING:
return Localization.instance.getString(asset.visibleName);
case TransactionEvent.STARBASE_RESEARCH:
out = "CodeString.FacebookShareResearch.Title";
localizationObj["[[String.ItemName]]"] = Localization.instance.getString(asset.visibleName);
localizationObj["[[String.Level]]"] = prototype.getUnsafeValue("level");
break;
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
out = "CodeString.FacebookShareUpgrade.Title";
localizationObj["[[String.ItemName]]"] = Localization.instance.getString(asset.visibleName);
localizationObj["[[String.Level]]"] = prototype.getUnsafeValue("level");
break;
}

if (out)
out = Localization.instance.getStringWithTokens(out, localizationObj);

return out;
}

static public function shareLevelUp( charName:String, lvl:int, isBaseLevel:Boolean ):void
{
if (ExternalInterface.available)
{
if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
var og:OpenGraphShareObject = new OpenGraphShareObject();
og.type = OpenGraphShareObject.ACTION_LEVELUP;
og.title = charName;
og.level = String(lvl);
//get the image
if (Application.ASSET_PATH != "" && Application.ASSET_PATH.indexOf('localhost') == -1)
og.image = Application.ASSET_PATH + "assets/" + ((isBaseLevel) ?
getSocialImage("BaseLevelup") : getSocialImage("Levelup"));
else
og.image = "http://oddalchemy.com/kabam/" + ((isBaseLevel) ? getSocialImage("BaseLevelup")
: getSocialImage("Levelup"));
//localize
var localizationObj:Dictionary = new Dictionary();
localizationObj["[[String.PlayerName]]"] = CurrentUser.name;
localizationObj["[[String.Level]]"] = lvl;
og.description = Localization.instance.getStringWithTokens(isBaseLevel ?
"CodeString.FacebookShareBaseLevelup" : "CodeString.FacebookShareLevelup",
localizationObj);
```

```
ExternalInterface.call(SHARE_USER_LEVEL_UP + "_facebook", og);
}
}
}

static public function shareAllianceJoin():void
{
if (ExternalInterface.available)
{
if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
var og:OpenGraphShareObject = new OpenGraphShareObject();
og.type = OpenGraphShareObject.ACTION_JOIN;
og.title = "Alliance";
if (Application.ASSET_PATH != "" && Application.ASSET_PATH.indexOf('localhost') == -1)
og.image = Application.ASSET_PATH + "assets/" + getSocialImage("Alliance");
else
og.image = "http://oddalchemy.com/kabam/" + getSocialImage("Alliance");
//localize
var localizationObj:Dictionary = new Dictionary();
localizationObj["[[String.PlayerName]]"] = CurrentUser.name;
localizationObj["[[String.AllianceName]]"] = CurrentUser.allianceName;
og.description =
Localization.instance.getStringWithTokens("CodeString.FacebookShareAlliance",
localizationObj);
ExternalInterface.call(SHARE_TRANSACTION + "_facebook", og);
}
}
}

static public function shareBlueprintFind( blueprint:IPrototype ):void
{
if (ExternalInterface.available)
{
if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
var asset:AssetVO = AssetModel.instance.getEntityData(blueprint.uiAsset);
if (asset)
{
var og:OpenGraphShareObject = new OpenGraphShareObject();
og.type = OpenGraphShareObject.ACTION_FIND;
og.title = Localization.instance.getString(asset.visibleName);
if (Application.ASSET_PATH != "" && Application.ASSET_PATH.indexOf('localhost') == -1)
og.image = Application.ASSET_PATH + "assets/" + getSocialImage("Blueprint", blueprint);
else
og.image = "http://oddalchemy.com/kabam/" + getSocialImage("Blueprint", blueprint);
//localize
var localizationObj:Dictionary = new Dictionary();
localizationObj["[[String.PlayerName]]"] = CurrentUser.name;
og.description
```

```
= Localization.instance.getStringWithTokens("CodeString.FacebookShareBlueprints",
localizationObj);
ExternalInterface.call(SHARE_TRANSACTION + "_facebook", og);
}
}
}
}

static private function getSocialImage( type:String, proto:IPrototype = null ):String
{
var img:String = "social/";
switch (type)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
case TransactionEvent.STARBASE_REFIT_BUILDING:
img += "BuildingComplete.png";
break;

case TransactionEvent.STARBASE_REFIT_SHIP:
img += "ShipRefit.png";
break;

case TransactionEvent.STARBASE_BUILD_SHIP:
img += "ShipResearchComplete.png";
break;

case TransactionEvent.STARBASE_RESEARCH:
img += (proto) ? "ShipResearchComplete.png" : "ResearchNewTech.png";
break;

case "Alliance":
img += "Leaderboard_Acc.png";
break;

case "BaseLevelup":
img += "BaseRating.png";
break;
case "Levelup":
img += "Milestone_UserLevels.png";
break;

case "BaseCombat":
img += "Destroyed_Tyr_Base.png";
break;
case "PVPCombat":
img += "WonPVPMatch.png";
break;

case "Blueprint":
var
```

```
rarity:String = proto.getValue('rarity');
if (rarity == 'Uncommon')
img += "Blueprint_Uncommon.png";
else if (rarity == 'Rare')
img += "Blueprint_Rare.png";
else if (rarity == 'Epic')
img += "Blueprint_Epic.png";
else if (rarity == 'Legendary')
img += "Blueprint_Legendary.png";
else
img += "Blueprint_Advanced.png";
break;
}
return img;
}


}
}


-----------------------------------
File 410: igw\com\service\OpenGraphShareObject.as
package com.service
{
import org.as3commons.logging.util.jsonXify;

/**
* This should closely map the generic Open Graph Object that we want to share via Facebook
and other implementations for Google+ and the like.
*/
internal class OpenGraphShareObject
{
static public const ACTION_BUILD:String = "build";
static public const ACTION_UPGRADE:String = "upgrade";
static public const ACTION_REFIT:String = "refit";
static public const ACTION_RESEARCH:String = "research";
static public const ACTION_LEVELUP:String = "levelup";
static public const ACTION_DEFEAT:String = "defeat";
static public const ACTION_JOIN:String = "join";
static public const ACTION_FIND:String = "find";

public var type:String;

public var url:String;
public var title:String;
public var image:String;

public var level:String;

public
```

```
var data:*;

public var description:String;
public var name:String;
public var caption:String;

public function toString():String
{
return jsonXify(this);
}
}
}

-------------------------------------
File 411: igw\com\service\facebook\FacebookAPI.as
package com.service.facebook
{
import com.service.ExternalInterfaceAPI;

public class FacebookAPI
{
private var _purchaseItemsCallback:Function;

public function purchaseCurrency( quantity:int, callback:Function ):void
{
_purchaseItemsCallback = callback;
ExternalInterfaceAPI.OpenFacebookPaywall(quantity);
}

public function onFacebookItemsPurchased( response:Object ):void
{
if (_purchaseItemsCallback != null)
{
_purchaseItemsCallback(response)
_purchaseItemsCallback = null;
}
}
}
}
-------------------------------------
File 412: igw\com\service\kongregate\KongregateAPI.as
package com.service.kongregate
{
import com.service.ExternalInterfaceAPI;

public class KongregateAPI
{
private var _purchaseItemsCallback:Function;

public function purchaseItems( items:Array, callback:Function ):void
{
```

```
_purchaseItemsCallback = callback;
ExternalInterfaceAPI.purchaseKongregateItems(items);
}

public function get gameAuthToken():String
{
return ExternalInterfaceAPI.getPlayerKongregateGameAuthToken();
}

public function get userName():String
{
return ExternalInterfaceAPI.getPlayerKongregateUsername();
}

public function get userID():Number
{
return ExternalInterfaceAPI.getPlayerKongregateUserId();
}

public function onKongregateItemsPurchased( response:Object ):void
{
if (_purchaseItemsCallback != null)
{
_purchaseItemsCallback(response)
_purchaseItemsCallback = null;
}
}
}
}

-------------------------------------
File 413: igw\com\service\language\Localization.as
package com.service.language
{
import com.Application;
import com.controller.ChatController;
import com.event.LoadEvent;
import com.model.asset.AssetModel;
import com.service.loading.LoadPriority;

import flash.events.Event;
import flash.events.IEventDispatcher;
import flash.net.SharedObject;
import flash.utils.Dictionary;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.osflash.signals.Signal;

import
```

```
com.service.ExternalInterfaceAPI;
public class Localization
{

public static var loaded:Boolean = false;
public static var instance:Localization;

[Inject]
public var assetModel:AssetModel;
[Inject]
public var chatController:ChatController;
[Inject]
public var eventDispatcher:IEventDispatcher;

public var onLoadFinished:Signal;

private var _defaultFallbackLanguage:String = 'en';
private var _language:String;
private var _loadingURL:String = 'data/localization/001_en.txt';
private var _stageWebserviceUrl:String = 'http://xlate-stage.kabam.com/index.php';
private var _strings:Dictionary;
private var _webserviceUrl:String = 'http://xlate.kabam.com/index.php';
public static var _languageEn:Boolean = false;
public static var _languageFr:Boolean = false;
public static var _languageEs:Boolean = false;
public static var _languageDe:Boolean = false;
public static var _languageIt:Boolean = false;
public static var _languagePl:Boolean = false;
private const _logger:ILogger = getLogger('LocalizationManager');

public function Localization()
{
instance = this;
_strings = new Dictionary();
onLoadFinished = new Signal();
}


private static var so:SharedObject = SharedObject.getLocal("IGWDATA");

//Toggle Language Function
public static function toggleLanguage( v:String ):void
{

//Save Language
switch (v){

case "en":
so.data.language = "en"
break;

case
```

```
"fr":
so.data.language = "fr"
break;

case "es":
so.data.language = "es"
break;

case "de":
so.data.language = "de"
break;

case "it":
so.data.language = "it"
break;

case "pl":
so.data.language = "pl"
break;

}
ExternalInterfaceAPI.setLocalizationTag(v);
ExternalInterfaceAPI.reloadSWF();
}


public function load( key:String = '', language:String = '' ):void
{
ExternalInterfaceAPI.logConsole("Language Tag: " + language);
var path:String = key;
_language = language;
if (language != '' && language != null)
path += '_' + language;
else
path += '_' + _defaultFallbackLanguage;
/*
if (Application.ASSET_PATH != null && Application.ASSET_PATH != '')
assetModel.getFromCache('xlate/' + path, onWebLoaded, LoadPriority.IMMEDIATE);
else
*/

// Onload download language files & set languageID to true
// Default English
_loadingURL = "data/localization/" + ExternalInterfaceAPI.getLocalizationFolder() + ".txt";


ExternalInterfaceAPI.logConsole("Language Folder: " + _loadingURL);

assetModel.getFromCache(_loadingURL, onloaded, LoadPriority.IMMEDIATE);
}

public
```

```
function onloaded( data:Object ):void
{
var strings:Array = data.locStrings;
var len:uint = strings.length;
for (var i:uint = 0; i < len; ++i)
{
addString(strings[i].trans, strings[i].phrase);
}

chatController.initStrings();
loaded = true;
onLoadFinished.dispatch();
eventDispatcher.dispatchEvent(new Event(LoadEvent.LOCALIZATION_COMPLETE));
}

public function onWebLoaded( data:Array ):void
{
if (data && data.length > 0)
{
for each (var locEntry:Object in data)
{
addString(locEntry.trans, locEntry.phrase);
}

chatController.initStrings();
loaded = true;
onLoadFinished.dispatch();
eventDispatcher.dispatchEvent(new Event(LoadEvent.LOCALIZATION_COMPLETE));
} else
assetModel.getFromCache(_loadingURL, onloaded, LoadPriority.IMMEDIATE);
}

private function addString( stringToAdd:String, key:String ):void
{
_strings[key] = stringToAdd;
}

public function getString( key:String ):String
{
var foundString:String;
if (key != "")
{
foundString = _strings[key];
if (foundString == null)
{
foundString = '';
}
} else
foundString = key;

return
```

```
foundString;
}

public function getStringWithTokens( key:String, tokens:Object ):String
{
var foundString:String;
if (key != "")
{
foundString = _strings[key];
if (foundString != null)
{
foundString = localizeStringWithTokens(foundString, tokens);
} else
{
foundString = '';
}
} else
foundString = key;

return foundString;
}

public function localizeStringWithTokens( stringWithSubs:String, tokens:Object ):String
{
var tokenString:String;
for (var key:String in tokens)
{

tokenString = getString(tokens[key]);
if (tokenString == '')
tokenString = tokens[key];

stringWithSubs = stringWithSubs.split(key).join(tokenString);
}
return stringWithSubs;
}



}
}

-----------------------------------
File 414: igw\com\service\loading\ILoadService.as
package com.service.loading
{
import com.service.loading.loaditems.ILoadItem;

public interface ILoadService
{
function cancel( loadItem:ILoadItem ):void;
function
```

```
lazyLoad( url:String, priority:int = 3, doLoad:Boolean = true, absoluteURL:Boolean = false
):ILoadItem;
function load( loadItem:ILoadItem ):void;
function loadBatch( type:int, urls:Array, priority:int = 3 ):ILoadItem;
function pause():void;
function reset():void;
function resume():void;

function get estimatedLoadCompleted():Number;
function get highPrioritiesInProgress():int;
function get highPrioritiesInWaiting():int;
function get highPrioritiesTotal():int;
function get itemsInWaiting():int;
}
}

-------------------------------------
File 415: igw\com\service\loading\LoadingTypes.as
package com.service.loading
{
public class LoadingTypes
{
public static const BITMAP:int = 0;

public static const SOUND:int = 1;

public static const SWF:int = 2;

public static const TEXT:int = 3;

public static const XML:int = 4;

public static const SPRITE_SHEET:int = 5;

public static const MESH:int = 6;

public static const SPRITE_SHEET_MESH:int = 7;

public static const BATTLEREPLAY:int = 8;
}
}

-------------------------------------
File 416: igw\com\service\loading\LoadMap.as
package com.service.loading
{

import com.service.loading.loaditems.ILoadItem;
import com.util.priorityqueue.ArrayPriorityQueue;
import com.util.priorityqueue.IPriorityQueue;

import
```

```
flash.utils.Dictionary;

internal final class LoadMap
{
//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

private var _highPrioritiesInProgress:int = 0;
private var _highPrioritiesInWaiting:int = 0;
private var _highPrioritiesTotal:int = 0;
private var _inProgressMap:Dictionary;
private var _inWaitingMap:Dictionary;
private var _loadsInProgress:int = 0;
private var _loadsWaiting:int = 0;
private var _queue:IPriorityQueue;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function LoadMap()
{
constructor();
}

private function constructor():void
{
_inProgressMap = new Dictionary(false);
_inWaitingMap = new Dictionary(false);
_queue = new ArrayPriorityQueue();
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public function add( loadItem:ILoadItem ):void
{
var url:String = loadItem.url;
if (_inWaitingMap[url] != null || _inProgressMap[url] != null)
return;
if (loadItem.priority < LoadPriority.MEDIUM)
{
_highPrioritiesInWaiting++;
_highPrioritiesTotal++;
}
_inWaitingMap[url] = loadItem;
_queue.add(loadItem);
_loadsWaiting++;
}
```

```
public function contains( loadItem:ILoadItem ):Boolean
{
if (_inProgressMap[loadItem.url] != null)
return true;
if (_inWaitingMap[loadItem.url] != null)
return true;
return false;
}

public function getNextLoadItem():ILoadItem
{
var next:ILoadItem = ILoadItem(_queue.getNext());
if (next)
{
remove(next);
if (_inProgressMap[next.url] == null)
{
_inProgressMap[next.url] = next;
_loadsInProgress++;
if (next.priority < LoadPriority.MEDIUM)
{
_highPrioritiesInProgress++;
_highPrioritiesInWaiting--;
}
}
}
return next;
}

public function remove( loadItem:ILoadItem ):void
{
var url:String = loadItem.url;
if (_inProgressMap[url] != null)
{
_loadsInProgress--;
if (loadItem.priority < LoadPriority.MEDIUM)
{
_highPrioritiesInProgress--;
_highPrioritiesTotal--;
}
}
if (_inWaitingMap[url] != null)
{
_loadsWaiting--;
_queue.remove(loadItem);
}
_inProgressMap[url] = null;
delete _inProgressMap[url];
_inWaitingMap[url]
```

```
= null;
delete _inWaitingMap[url];
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public function get highPrioritiesInProgress():int { return _highPrioritiesInProgress; }
public function get highPrioritiesInWaiting():int { return _highPrioritiesInWaiting; }
public function get highPrioritiesTotal():int { return _highPrioritiesTotal; }
public function get loadsInProgress():int { return _loadsInProgress; }
public function get loadsWaiting():int { return _loadsWaiting; }
}
}


-------------------------------------
File 417: igw\com\service\loading\LoadPriority.as
package com.service.loading
{

public final class LoadPriority
{

//////////////////////////////////////////////////////////
// CONSTANTS
//////////////////////////////////////////////////////////

public static const IMMEDIATE:int = 0;

public static const HIGH:int = 1;

public static const MEDIUM:int = 2;

public static const LOW:int = 3;

public static const DONT_GIVE_A_SHIT:int = 4;

}
}
-------------------------------------
File 418: igw\com\service\loading\LoadService.as
package com.service.loading
{

import com.Application;
import com.enum.TimeLogEnum;
import com.event.LoadEvent;
import com.service.loading.loaditems.BatchLoadItem;
import
```

```
com.service.loading.loaditems.ILoadItem;
import com.service.loading.loaditems.LoadItem;
import com.service.loading.loaditems.LoaderLoadItem;
import com.service.loading.loaditems.SoundLoadItem;
import com.service.loading.loaditems.URLLoaderLoadItem;
import com.util.TimeLog;

import flash.events.Event;
import flash.events.IEventDispatcher;
import flash.events.ProgressEvent;
import flash.utils.getTimer;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

/**
*
* USAGE:
*
* var loadService:ILoadService = new LoadService();
*
* var loadItem:ILoadItem;
*
* loadItem = new BitmapLoadItem("http://server/image.png", LoadPriority.HIGH);
* loadItem = new SWFLoadItem("http://server/image.png", LoadPriority.MEDIUM);
* loadItem = new SoundLoadItem("http://server/image.png", LoadPriority.LOW);
* loadItem = new XMLLoadItem("http://server/image.png", LoadPriority.PRELOAD);
* loadItem = new TextLoadItem("http://server/image.png", LoadPriority.IMMEDIATE);
*
* loadItem.priority = LoadPriority.IMMEDIATE;
* loadItem.url = "http://server/asset.jpg";
* trace(loadItem.progress); // 0.0 - 1.0
*
* loadItem.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
* loadItem.addEventListener(LoadEvent.COMPLETE, onLoadComplete);
* loadItem.addEventListener(LoadEvent.START, onLoadStart);
* loadItem.addEventListener(LoadProgressEvent.PROGRESS, onLoadProgress);
*
* loadService.load(loadItem);
* loadService.cancel(loadItem);
*
* var bitmap:Bitmap = Bitmap(bitmapLoadItem.asset); // will be null until LoadEvent.COMPLETE
is fired
* bitmap = bitmapLoadItem.bitmap; // will be null until LoadEvent.COMPLETE is fired
*
* var displayObject:DisplayObject = DisplayObject(swfLoadItem.asset); // will be null until
LoadEvent.COMPLETE is fired
* displayObject = swfLoadItem.displayObject; // will be null until LoadEvent.COMPLETE is fired
*
* var sound:Sound = Sound(soundLoadItem.asset); // will be null until LoadEvent.COMPLETE is
fired
```

```
* sound = soundLoadItem.sound; // will be null until LoadEvent.COMPLETE is fired
*
* var xml:XML = XML(xmlLoadItem.asset); // will be null until LoadEvent.COMPLETE is fired
* xml = xmlLoadItem.xml; // will be null until LoadEvent.COMPLETE is fired
*
* var text:String = String(textLoadItem.asset); // will be null until LoadEvent.COMPLETE is fired
* text = textLoadItem.text; // will be null until LoadEvent.COMPLETE is fired
*
*/
public final class LoadService implements ILoadService
{

////////////////////////////////////////////////////////
// CONSTANTS
////////////////////////////////////////////////////////

public static const ALL_COMPLETE:String = "AllComplete";
private static const MAX_CONNECTIONS:int = 5;
private const _logger:ILogger = getLogger('LoadService');

////////////////////////////////////////////////////////
// ATTRIBUTES
////////////////////////////////////////////////////////

private var _dispatcher:IEventDispatcher;
private var _itemsLoaded:int = 0;
private var _loadMap:LoadMap;
private var _loadTimes:Vector.<int>;
private var _paused:Boolean = false;

//estimated loading
private var _estimatedTimeToFinish:Number = 0;
private var _fileSizeAverage:int = 1000000;
private var _totalBytesLoaded:int = 0;
private var _totalBytesLoading:int = 0;

////////////////////////////////////////////////////////
// CONSTRUCTOR
////////////////////////////////////////////////////////

[PostConstruct]
public function constructor():void
{
_loadMap = new LoadMap();
_loadTimes = new Vector.<int>();
}

////////////////////////////////////////////////////////
// PUBLIC API
////////////////////////////////////////////////////////
```

```
public function cancel( loadItem:ILoadItem ):void
{
loadItem.cancel();
_loadMap.remove(loadItem);
loadNext();
}

public function lazyLoad( url:String, priority:int = 3, doLoad:Boolean = true,
absoluteURL:Boolean = false ):ILoadItem
{
var loader:ILoadItem;
var index:int = url.lastIndexOf('.');
var extension:String = (index != -1) ? url.substr(index + 1).toLowerCase() : '';

switch (extension)
{
/*case '3ds':
loader = new Away3DLoadItem(url, LoadingTypes.MESH, priority);
break;*/
case 'jpg':
case 'jpeg':
case 'png':
case 'gif':
loader = new LoaderLoadItem(url, LoadingTypes.BITMAP, priority);
break;
case 'wav':
case 'mp3':
loader = new SoundLoadItem(url, LoadingTypes.SOUND, priority);
break;
case 'swf':
loader = new LoaderLoadItem(url, LoadingTypes.SWF, priority);
break;
case 'xml':
loader = new URLLoaderLoadItem(url, LoadingTypes.XML, priority);
break;
case 'battle':
loader = new URLLoaderLoadItem(url, LoadingTypes.BATTLEREPLAY, priority);
break;
default:
loader = new URLLoaderLoadItem(url, LoadingTypes.TEXT, priority);
break;
}
if (doLoad)
load(loader);
return loader;
}

public function load( loadItem:ILoadItem ):void
{
```

```
loadItem.prefix = (loadItem.absoluteURL) ? '' : Application.ASSET_PATH;
// look to see if that asset is already in progress
if (!_loadMap.contains(loadItem))
_loadMap.add(loadItem);

loadNext();
}

public function loadBatch( type:int, urls:Array, priority:int = 3 ):ILoadItem
{
var batchLoader:BatchLoadItem = new BatchLoadItem(urls, type, priority);
load(batchLoader);
for (var i:int = 0; i < urls.length; i++)
batchLoader.addLoadItem(lazyLoad(urls[i], priority, false));
for (i = 0; i < batchLoader.items.length; i++)
load(batchLoader.items[i]);
return batchLoader;
}

public function pause():void
{
//doesn't stop any current loads it just prevents any more loads from this point on
_paused = true;
}

public function resume():void
{
_paused = false;
loadNext();
}

public function reset():void
{
_estimatedTimeToFinish = 0;
_totalBytesLoaded = 0;
_totalBytesLoading = 1;
}

////////////////////////////////////////////////////////////
// PRIVATE METHODS
////////////////////////////////////////////////////////////

private function addLoadItemListeners( loadItem:ILoadItem ):void
{
loadItem.addUpdateListener(onUpdate);
}

private function removeLoadItemListeners( loadItem:ILoadItem ):void
{
loadItem.removeUpdateListener(onUpdate);
```

```
}

private function onUpdate( state:String, loadItem:ILoadItem ):void
{
switch (state)
{
case LoadItem.CANCEL:
cancel(loadItem);
break;
case LoadItem.COMPLETE:
var endTime:int = getTimer();
if (loadItem.type != LoadingTypes.SPRITE_SHEET || loadItem.type !=
LoadingTypes.SPRITE_SHEET_MESH)
{
TimeLog.endTimeLog(TimeLogEnum.FILE_LOAD, loadItem.url);
_logger.info('Loading Complete: {0} Time: {1}', [loadItem.fullPath, endTime -
loadItem.loadStartTime]);
updateAvgLoadTime(loadItem.totalBytesLoaded, loadItem.loadStartTime, endTime);
}
removeLoadItemListeners(loadItem);
_loadMap.remove(loadItem);
_dispatcher.dispatchEvent(new LoadEvent(loadItem));

_itemsLoaded++;

var progressEvent:ProgressEvent = new ProgressEvent(ProgressEvent.PROGRESS);
progressEvent.bytesLoaded = _itemsLoaded;
progressEvent.bytesTotal = _loadMap.loadsInProgress + _loadMap.loadsWaiting +
_itemsLoaded;
_dispatcher.dispatchEvent(progressEvent);
loadNext();
break;
case LoadItem.PROGRESS:
if (loadItem.priority < LoadPriority.MEDIUM)
{
if (!loadItem.tracked)
{
_totalBytesLoading += loadItem.totalBytes;
loadItem.tracked = true;
}
_totalBytesLoaded += loadItem.totalBytesLoadedLastFrame;
//trace(estimatedTimeToFinish, estimatedLoadCompleted);
}
break;
case LoadItem.START:
break;
default:
_logger.error("Load error: {0}, {1}", [state, loadItem.fullPath]);
_loadMap.remove(loadItem);
loadNext();
```

```
break;
}
}

private function loadNext():void
{
// stop if we're already loading too many
if (_loadMap.loadsInProgress >= MAX_CONNECTIONS || _paused)
{
return;
}

// stop if there's nothing to load
if (_loadMap.loadsWaiting == 0)
{
//We've finished loading all assets...?
if (_loadMap.loadsInProgress == 0)
{
_itemsLoaded = 0;
_dispatcher.dispatchEvent(new Event(ALL_COMPLETE));
}
return;
}

// get the next item to load
var loadItem:ILoadItem = _loadMap.getNextLoadItem();
// listen to the "master" load item for this URL
addLoadItemListeners(loadItem);

_logger.info('Loading: {}', loadItem.fullPath);
loadItem.load();

if (_loadMap.loadsInProgress < MAX_CONNECTIONS && _loadMap.loadsWaiting > 0)
loadNext();
}

private function updateAvgLoadTime( bytesLoaded:Number, startTime:Number,
endTime:Number ):void
{
if ((endTime - startTime) <= 0)
return;
_loadTimes.push(bytesLoaded / (endTime - startTime));
if (_loadTimes.length > 10)
_loadTimes.shift();

var len:uint = _loadTimes.length;
if (len > 2)
{
var avgLoadTime:int = 0;
for
```

```
(var i:uint = 0; i < len; ++i)
{
avgLoadTime += _loadTimes[i];
}

avgLoadTime /= len;
Application.AVG_LOAD_TIME = avgLoadTime;
}
}

public function get estimatedLoadCompleted():Number { return _totalBytesLoaded /
(_totalBytesLoading + (_loadMap.highPrioritiesInWaiting * _fileSizeAverage)); }
public function get highPrioritiesInProgress():int { return _loadMap.highPrioritiesInProgress; }
public function get highPrioritiesInWaiting():int { return _loadMap.highPrioritiesInWaiting; }
public function get highPrioritiesTotal():int { return _loadMap.highPrioritiesTotal; }
public function get itemsInWaiting():int { return _loadMap.loadsWaiting; }

[Inject]
public function set dispatcher( v:IEventDispatcher ):void { _dispatcher = v; }
}
}

-------------------------------------
File 419: igw\com\service\loading\URL.as
package com.service.loading
{
/**
* Supports:
*
* http://localhost/path/file.php
* localhost/path/file.php?query=String&param=val
* file:///some/path/file.php
*/
public final class URL
{

//////////////////////////////////////////////////////////
// CONSTANTS
//////////////////////////////////////////////////////////

public static const FILE_PROTOCOL:String = "file";

public static const HTTP_PROTOCOL:String = "http";

//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

public var prefix:String = '';

private
```

```
var _url:String = "";

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function URL( url:String )
{
_url = url;
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public function toString():String
{
return prefix + _url;
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

// rename: fullPath?
public function get baseUrl():String
{
var end:int = _url.indexOf("?");

if (end == -1)
{
end = _url.length;
}

return _url.substring(0, end);
}

public function get domain():String
{
//TODO: URL.domain
return "";
}

public function get filename():String
{
var start:int = _url.lastIndexOf("/") + 1;
var end:int = _url.indexOf("?");

if (end == -1)
{
end
```

```
= _url.length;
}

return _url.substring(start, end);
}

public function get parameters():Object
{
var params:Object = new Object();
var qs:String = queryString;

if (qs == "")
{
return params;
}

var pairs:Array = qs.split("&");

for each (var pair:String in pairs)
{
var keyVal:Array = pair.split("=");
var key:String = keyVal[0];
var val:String = keyVal[1];
params[key] = val;
}

return params;
}

public function get path():String
{
var endOfProtocol:int = _url.lastIndexOf("//") + 2;
var startOfPath:int = _url.indexOf("/", endOfProtocol);
var endOfPath:int = _url.lastIndexOf("/") + 1;

return _url.substring(startOfPath, endOfPath);
}

public function get port():int
{
var endOfProtocol:int = _url.lastIndexOf("//") + 2;
var startOfPort:int = _url.indexOf(":", endOfProtocol) + 1;

if (startOfPort == 0)
{
return 80;
}

var endOfPort:int = _url.indexOf("/", startOfPort);
return int(_url.substring(startOfPort, endOfPort));
}
```

```
public function get protocol():String
{
var index:int = _url.indexOf("//");

if (index == -1)
{
return HTTP_PROTOCOL;
}

return _url.substring(0, index - 1);
}

public function get queryString():String
{
var start:int = _url.indexOf("?") + 1;

if (start == 0)
{
start = _url.length;
}

return _url.substring(start);
}

public function get server():String
{
var endOfProtocol:int = _url.lastIndexOf("//");

if (endOfProtocol == -1)
{
endOfProtocol = 0;
} else
{
endOfProtocol += "//".length;
}

var startOfPath:int = _url.indexOf("/", endOfProtocol);
var startOfPort:int = _url.indexOf(":", endOfProtocol);

if (startOfPath == -1)
{
startOfPath = _url.length;
}

if (startOfPort == -1)
{
startOfPort = _url.length;
}

var
```

```
endOfServer:int = Math.min(startOfPath, startOfPort);

return _url.substring(endOfProtocol, endOfServer);
}
}
}

-----------------------------------
File 420: igw\com\service\loading\loaditems\Away3DLoadItem.as
package com.service.loading.loaditems
{

import flash.errors.IOError;
import flash.net.URLRequest;
import flash.system.ApplicationDomain;
import flash.system.LoaderContext;

import org.away3d.events.AssetEvent;
import org.away3d.library.assets.AssetType;
import org.away3d.loaders.Loader3D;

/**
* This is the internal base class for LoadItems that use a Loader
* to download their asset
*/
public class Away3DLoadItem extends LoadItem
{
private static const _context:LoaderContext = new LoaderContext(false,
ApplicationDomain.currentDomain);

protected var _loader:Loader3D;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function Away3DLoadItem( url:String, type:int, priority:int = int.MAX_VALUE,
absolute:Boolean = false )
{
super(url, type, priority, absolute);
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public override function cancel():void
{
if (_loader)
{
removeLoaderListeners(_loader);
```

```
try
{
_loader.stopLoad();
_loader = null;
} catch ( e:IOError )
{
// The loader will throw an IOError (with error code 2029)
// if it wasn't loading anything, which we ignore. If it was
// some other kind of error, we throw the error
if (e.errorID != 2029)
{
throw e;
}
}
}

_updateSignal.dispatch(CANCEL, this);
}

public override function destroy():void
{
super.destroy();

if (_loader)
{
removeLoaderListeners(_loader);
_loader = null;
}
}

public override function load():void
{
if (!_loader)
{
_loader = new Loader3D(false);
addLoaderListeners(_loader);
}
_loader.load(new URLRequest(_url.toString()));
}

protected function onAssetLoadComplete( event:AssetEvent ):void
{
if (event.asset.assetType == AssetType.MESH)
{
this.asset = event.asset;
_loaded = true;
_updateSignal.dispatch(COMPLETE, this);
}
}
```

```
//////////////////////////////////////////////////////////
// PRIVATE METHODS
//////////////////////////////////////////////////////////

private function addLoaderListeners( loader:Loader3D ):void
{
loader.addEventListener(AssetEvent.ASSET_COMPLETE, onAssetLoadComplete);
}

private function removeLoaderListeners( loader:Loader3D ):void
{
loader.removeEventListener(AssetEvent.ASSET_COMPLETE, onAssetLoadComplete);
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public override function get progress():Number
{
if (!_loader)
{
return 0;
}
return 0; //_loader.contentLoaderInfo.bytesLoaded / _loader.contentLoaderInfo.bytesTotal;
}
}
}

-----------------------------------
File 421: igw\com\service\loading\loaditems\BatchLoadItem.as
package com.service.loading.loaditems
{


public final class BatchLoadItem extends LoadItem
{

//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

public var category:String;
public var entityType:String;

private var _items:Vector.<ILoadItem>;
private var _itemsComplete:int;

//////////////////////////////////////////////////////////
```

```
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function BatchLoadItem( urls:Array, type:int, priority:int = 0, absolute:Boolean = false )
{
_items = new Vector.<ILoadItem>;
_itemsComplete = 0;
super(urls.join(',') + "Batch", type, priority, absolute);
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public function addLoadItem( loadItem:ILoadItem ):void
{
_items.push(loadItem);
addLoadListeners(loadItem);
}

//do nothing
public override function load():void
{

}

//////////////////////////////////////////////////////////
// PRIVATE METHODS
//////////////////////////////////////////////////////////

private function addLoadListeners( loadItem:ILoadItem ):void
{
loadItem.addUpdateListener(onUpdate);
}

private function removeLoadListeners( loadItem:ILoadItem ):void
{
loadItem.removeUpdateListener(onUpdate);
}

private function onUpdate( state:String, loadItem:ILoadItem ):void
{
switch (state)
{
case LoadItem.COMPLETE:
removeLoadListeners(loadItem);
_itemsComplete++;
if (_itemsComplete == _items.length)
_updateSignal.dispatch(LoadItem.COMPLETE, this);
break;
```

```
}
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public function get items():Vector.<ILoadItem> { return _items; }

public override function get progress():Number
{
return _itemsComplete / _items.length;
}
}
}

-------------------------------------
File 422: igw\com\service\loading\loaditems\BattleReplayLoadItem.as
package com.service.loading.loaditems
{
import flash.errors.IOError;
import flash.events.Event;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.net.URLRequest;

public final class BattleReplayLoadItem extends LoadItem
{

//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

//protected var _loader:Sound;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function BattleReplayLoadItem( url:String, type:int, priority:int = int.MAX_VALUE,
absolute:Boolean = false )
{
super(url, type, priority, absolute);
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public
```

```
override function cancel():void
{
if (_loader)
{
removeLoaderListeners(_loader);

try
{
_loader.close();
} catch ( e:IOError )
{
// The loader will throw an IOError (with error code 2029)
// if it wasn't loading anything, which we ignore. If it was
// some other kind of error, we throw the error
if (e.errorID != 2029)
{
throw e;
}
}
}

_updateSignal.dispatch(CANCEL, this);
}

public override function destroy():void
{
super.destroy();

if (_loader)
{
removeLoaderListeners(_loader);
_loader = null;
}
}

public override function load():void
{
if (_loader)
removeLoaderListeners(_loader);

_loader = new Sound();
addLoaderListeners(_loader);
_loader.load(new URLRequest(_url.toString()));
}

//////////////////////////////////////////////////////////
// PRIVATE METHODS
//////////////////////////////////////////////////////////

private function addLoaderListeners( loader:Sound ):void
{
```

```
loader.addEventListener(Event.OPEN, onLoadStart);
loader.addEventListener(ProgressEvent.PROGRESS, onLoadProgress);
loader.addEventListener(Event.COMPLETE, onLoadComplete);
loader.addEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

private function removeLoaderListeners( loader:Sound ):void
{
_loader.removeEventListener(Event.OPEN, onLoadStart);
_loader.removeEventListener(ProgressEvent.PROGRESS, onLoadProgress);
_loader.removeEventListener(Event.COMPLETE, onLoadComplete);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public override function get progress():Number
{
if (!_loader)
{
return 0;
}

return _loader.bytesLoaded / _loader.bytesTotal;
}

// protected override function onLoadComplete(event:Event):void
// {
// this.asset = Bitmap(_loader.content);
// }

public function get sound():Sound
{
if (_asset)
{
return Sound(_asset);
}

return null;
}
}
}

-----------------------------------
File 423: igw\com\service\loading\loaditems\ILoadItem.as
package com.service.loading.loaditems
{

import
```

```
com.util.priorityqueue.IPrioritizable;

import flash.system.ApplicationDomain;

public interface ILoadItem extends IPrioritizable
{
function cancel():void;
function load():void;

function addUpdateListener( callback:Function ):void;
function removeUpdateListener( callback:Function ):void;

function get absoluteURL():Boolean;

function get applicationDomain():ApplicationDomain;

function get asset():Object;
function set asset( asset:Object ):void;

function get filename():String;

function get loaded():Boolean;

function set prefix( v:String ):void;

function get progress():Number;

function get type():int;

function get url():String;

function get fullPath():String;

function get loadStartTime():int;

function get totalBytes():int;
function get totalBytesLoaded():int;
function get totalBytesLoadedLastFrame():int;

function get tracked():Boolean;
function set tracked( v:Boolean ):void;
}
}

-----------------------------------
File 424: igw\com\service\loading\loaditems\LoaderLoadItem.as
package com.service.loading.loaditems
{

import flash.display.Loader;
import
```

```
flash.errors.IOError;
import flash.events.Event;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.net.URLRequest;
import flash.system.ApplicationDomain;
import flash.system.LoaderContext;

/**
* This is the internal base class for LoadItems that use a Loader
* to download their asset
*/
public class LoaderLoadItem extends LoadItem
{
private static const _context:LoaderContext = new LoaderContext(false,
ApplicationDomain.currentDomain);

//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

protected var _loader:Loader;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function LoaderLoadItem( url:String, type:int, priority:int = int.MAX_VALUE,
absolute:Boolean = false )
{
super(url, type, priority, absolute);
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public override function cancel():void
{
if (_loader)
{
removeLoaderListeners(_loader);

try
{
_loader.close();
} catch ( e:IOError )
{
// The loader will throw an IOError (with error code 2029)
// if it wasn't loading anything, which we ignore. If it was
//
```

```
some other kind of error, we throw the error
if (e.errorID != 2029)
{
throw e;
}
}
}

_updateSignal.dispatch(CANCEL, this);
}

public override function destroy():void
{
super.destroy();

if (_loader)
{
removeLoaderListeners(_loader);
_loader = null;
}
}

public override function load():void
{
// ensure that only 1 loader is used for this BitmapLoadItem instance
if (!_loader)
{
_loader = new Loader();
addLoaderListeners(_loader);
}
_loader.load(new URLRequest(_url.toString()), _context);
}

override protected function onLoadComplete( event:Event ):void
{
_applicationDomain = _loader.contentLoaderInfo.applicationDomain;
super.onLoadComplete(event);
}

//////////////////////////////////////////////////////////
// PRIVATE METHODS
//////////////////////////////////////////////////////////

private function addLoaderListeners( loader:Loader ):void
{
loader.contentLoaderInfo.addEventListener(Event.OPEN, onLoadStart);
loader.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS, onLoadProgress);
loader.contentLoaderInfo.addEventListener(Event.COMPLETE, onLoadComplete);
loader.contentLoaderInfo.addEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

private
```

```
function removeLoaderListeners( loader:Loader ):void
{
_loader.contentLoaderInfo.removeEventListener(Event.OPEN, onLoadStart);
_loader.contentLoaderInfo.removeEventListener(ProgressEvent.PROGRESS,
onLoadProgress);
_loader.contentLoaderInfo.removeEventListener(Event.COMPLETE, onLoadComplete);
_loader.contentLoaderInfo.removeEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public override function get progress():Number
{
if (!_loader)
{
return 0;
}

return _loader.contentLoaderInfo.bytesLoaded / _loader.contentLoaderInfo.bytesTotal;
}
}
}

-----------------------------------
File 425: igw\com\service\loading\loaditems\LoadItem.as
package com.service.loading.loaditems
{

import com.enum.TimeLogEnum;
import com.service.loading.LoadingTypes;
import com.service.loading.URL;
import com.util.TimeLog;
import com.util.priorityqueue.IPrioritizable;

import flash.display.LoaderInfo;
import flash.events.Event;
import flash.events.EventDispatcher;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.media.Sound;
import flash.net.URLLoader;
import flash.system.ApplicationDomain;
import flash.utils.getTimer;

import org.osflash.signals.Signal;

public class LoadItem extends EventDispatcher implements ILoadItem, IPrioritizable
{
```

```
//////////////////////////////////////////////////////////
// CONSTANTS
//////////////////////////////////////////////////////////

public static const CANCEL:String = "cancel";

public static const COMPLETE:String = "complete";

public static const START:String = "start";

public static const ERROR:String = "error";

public static const PROGRESS:String = "progress";

public static const MAX_RETRIES:uint = 3;

//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

protected var _absoluteURL:Boolean;
protected var _applicationDomain:ApplicationDomain;
protected var _asset:Object;
protected var _loaded:Boolean;
protected var _loadStartTime:int;
protected var _priority:int;
protected var _retries:int;
protected var _tracked:Boolean;
protected var _type:int;
protected var _updateSignal:Signal;
protected var _url:URL;
protected var _totalBytes:int;
protected var _totalBytesLoaded:int;
protected var _totalBytesLoadedLastFrame:int;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function LoadItem( url:String, type:int, priority:int = int.MAX_VALUE, absolute:Boolean =
false )
{
_absoluteURL = absolute;
_loaded = false;
_priority = priority;
_retries = 0;
_totalBytes = 0;
_totalBytesLoaded = 0;
_totalBytesLoadedLastFrame = 0;
_tracked
```

```
= false;
_type = type;
_updateSignal = new Signal(String, ILoadItem);
_url = new URL(url);
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public function cancel():void
{
throw new Error("LoadItem.cancel() should be implemented by a subclass");
}

public function destroy():void
{
_url = null;
_asset = null;
_updateSignal.removeAll();
_updateSignal = null;
}

public function load():void
{
throw new Error("LoadItem.load() should be implemented by a subclass");
}

public function addUpdateListener( callback:Function ):void
{
_updateSignal.add(callback);
}

public function removeUpdateListener( callback:Function ):void
{
_updateSignal.remove(callback);
}

//////////////////////////////////////////////////////////
// PROTECTED METHODS
//////////////////////////////////////////////////////////

protected function onLoadComplete( event:Event ):void
{
var loader:Object = event.target;
_loaded = true;

if (loader is LoaderInfo)
{
this.asset = LoaderInfo(loader).content;
}
```

```
else if (loader is Sound)
{
this.asset = loader;
} else if (loader is URLLoader)
{
this.asset = URLLoader(loader).data;
}

_updateSignal.dispatch(COMPLETE, this);
}

protected function onLoadIOError( event:IOErrorEvent ):void
{
_retries++;
if (_retries < MAX_RETRIES)
load();
else
_updateSignal.dispatch(ERROR + event.text, this);
}

protected function onHTTPStatusError( event:HTTPStatusEvent ):void
{
_retries++;
if (_retries < MAX_RETRIES)
load();
else
_updateSignal.dispatch(ERROR, this);
}

protected function onLoadProgress( event:Event ):void
{
if (_totalBytes == 0)
_totalBytes = ProgressEvent(event).bytesTotal;
_totalBytesLoadedLastFrame = ProgressEvent(event).bytesLoaded - _totalBytesLoaded;
_totalBytesLoaded = ProgressEvent(event).bytesLoaded;
_updateSignal.dispatch(PROGRESS, this);
}

protected function onLoadStart( event:Event ):void
{
_updateSignal.dispatch(START, this);
if (_type != LoadingTypes.SPRITE_SHEET || type != LoadingTypes.SPRITE_SHEET_MESH)
{
TimeLog.startTimeLog(TimeLogEnum.FILE_LOAD, _url.baseUrl);
_loadStartTime = getTimer();
}
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////
```

```
public function get absoluteURL():Boolean
{
return _absoluteURL;
}

public function get applicationDomain():ApplicationDomain
{
return _applicationDomain;
}

public function get asset():Object
{
return _asset;
}

public function set asset( value:Object ):void
{
_asset = value;
}

public function get filename():String
{
return _url.filename;
}

public function get loaded():Boolean
{
return _loaded;
}

public function set prefix( v:String ):void
{
_url.prefix = v;
}

public function get priority():int
{
return _priority;
}

public function get progress():Number
{
throw new Error("LoadItem.get progress() should be implemented by a subclass");
}

public function get type():int { return _type; }

public function get url():String
{
```

```
var query:String = _url.queryString;
if( query == '' )
{
return _url.baseUrl;
}
else
{
return _url.baseUrl + "?" + _url.queryString;
}
}

public function get fullPath():String
{
return _url.toString();
}

public function get loadStartTime():int
{
return _loadStartTime;
}

public function get totalBytes():int { return _totalBytes; }
public function get totalBytesLoaded():int { return _totalBytesLoaded; }
public function get totalBytesLoadedLastFrame():int { return _totalBytesLoadedLastFrame; }

public function get tracked():Boolean { return _tracked; }
public function set tracked( v:Boolean ):void { _tracked = v; }
}
}

-----------------------------------
File 426: igw\com\service\loading\loaditems\SoundLoadItem.as
package com.service.loading.loaditems
{
import com.service.ExternalInterfaceAPI;

import flash.errors.IOError;
import flash.events.Event;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.media.Sound;
import flash.net.URLRequest;

public final class SoundLoadItem extends LoadItem
{

//////////////////////////////////////////////////////////
// ATTRIBUTES
//////////////////////////////////////////////////////////

static
```

```
private const LOCALIZATION_FOLDER_SEGMENT:String = "LOCALIZATION_SEGMENT";
static private const LOCALIZATION_VO_FOLDER_SEGMENT:String =
"LOCALIZATION_VO_SEGMENT";
static private const LOCALIZATION_FOLDER_PATTERN:RegExp =
/LOCALIZATION_SEGMENT/;
static private const LOCALIZATION_VO_FOLDER_PATTERN:RegExp =
/LOCALIZATION_VO_SEGMENT/;

protected var _loader:Sound;

//////////////////////////////////////////////////////////
// CONSTRUCTOR
//////////////////////////////////////////////////////////

public function SoundLoadItem( url:String, type:int, priority:int = int.MAX_VALUE,
absolute:Boolean = false )
{
super(url, type, priority, absolute);
}

//////////////////////////////////////////////////////////
// PUBLIC API
//////////////////////////////////////////////////////////

public override function cancel():void
{
if (_loader)
{
removeLoaderListeners(_loader);

try
{
_loader.close();
} catch ( e:IOError )
{
// The loader will throw an IOError (with error code 2029)
// if it wasn't loading anything, which we ignore. If it was
// some other kind of error, we throw the error
if (e.errorID != 2029)
{
throw e;
}
}
}

_updateSignal.dispatch(CANCEL, this);
}

public override function destroy():void
{
super.destroy();
```

```
if (_loader)
{
removeLoaderListeners(_loader);
_loader = null;
}
}

public override function load():void
{
if (_loader)
removeLoaderListeners(_loader);

_loader = new Sound();
addLoaderListeners(_loader);
_loader.load(new URLRequest(parseUrlToLoad(_url.toString())));
}

//////////////////////////////////////////////////////////
// PRIVATE METHODS
//////////////////////////////////////////////////////////

private function parseUrlToLoad(urlToLoad:String):String
{
var index:int = urlToLoad.indexOf(LOCALIZATION_FOLDER_SEGMENT);
if(index!=-1)
{
var localizationFolder:String = ExternalInterfaceAPI.getLocalizationFolder();
if(localizationFolder.length>0)
{
urlToLoad = urlToLoad.replace(LOCALIZATION_FOLDER_PATTERN,localizationFolder);
ExternalInterfaceAPI.logConsole("ParseURL: " + urlToLoad);
}
return urlToLoad;
}

index = urlToLoad.indexOf(LOCALIZATION_VO_FOLDER_SEGMENT);
if(index!=-1)
{
var localizationVOFolder:String = ExternalInterfaceAPI.getLocalizationVOFolder();
if(localizationVOFolder.length>0)
{
urlToLoad =
urlToLoad.replace(LOCALIZATION_VO_FOLDER_PATTERN,localizationVOFolder);
ExternalInterfaceAPI.logConsole("ParseURL: " + urlToLoad);
}
return urlToLoad;
}

return
```

```
urlToLoad;
}

private function addLoaderListeners( loader:Sound ):void
{
loader.addEventListener(Event.OPEN, onLoadStart);
loader.addEventListener(ProgressEvent.PROGRESS, onLoadProgress);
loader.addEventListener(Event.COMPLETE, onLoadComplete);
loader.addEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

private function removeLoaderListeners( loader:Sound ):void
{
_loader.removeEventListener(Event.OPEN, onLoadStart);
_loader.removeEventListener(ProgressEvent.PROGRESS, onLoadProgress);
_loader.removeEventListener(Event.COMPLETE, onLoadComplete);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public override function get progress():Number
{
if (!_loader)
{
return 0;
}

return _loader.bytesLoaded / _loader.bytesTotal;
}

// protected override function onLoadComplete(event:Event):void
// {
// this.asset = Bitmap(_loader.content);
// }

public function get sound():Sound
{
if (_asset)
{
return Sound(_asset);
}

return null;
}
}
}
```

File 427: igw\com\service\loading\loaditems\URLLoaderLoadItem.as

```
package com.service.loading.loaditems
{

import com.service.loading.LoadingTypes;

import flash.errors.IOError;
import flash.events.Event;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.net.URLLoader;
import flash.net.URLLoaderDataFormat;
import flash.net.URLRequest;

public class URLLoaderLoadItem extends LoadItem
{

////////////////////////////////////////////////////////
// ATTRIBUTES
////////////////////////////////////////////////////////

protected var _loader:URLLoader;

////////////////////////////////////////////////////////
// CONSTRUCTOR
////////////////////////////////////////////////////////

public function URLLoaderLoadItem( url:String, type:int, priority:int = int.MAX_VALUE,
absolute:Boolean = false )
{
super(url, type, priority, absolute);
}

////////////////////////////////////////////////////////
// PUBLIC API
////////////////////////////////////////////////////////

public override function cancel():void
{
if (_loader)
{
removeLoaderListeners(_loader);

try
{
_loader.close();
} catch ( e:IOError )
{
// The loader will throw an IOError (with error code 2029)
//
```

```
if it wasn't loading anything, which we ignore. If it was
// some other kind of error, we throw the error
if (e.errorID != 2029)
{
throw e;
}
}
}

_updateSignal.dispatch(CANCEL, this);
}

public override function destroy():void
{
super.destroy();

if (_loader)
{
removeLoaderListeners(_loader);
_loader = null;
}
}

public override function load():void
{
// ensure that only 1 loader is used for this BitmapLoadItem instance
if (!_loader)
{
_loader = new URLLoader();
addLoaderListeners(_loader);
}

//TODO: ??? supply LoaderContext to separate application domains
if( _type == LoadingTypes.BATTLEREPLAY )
{
_loader.dataFormat = URLLoaderDataFormat.BINARY;
}
_loader.load(new URLRequest(_url.toString()));
}

//////////////////////////////////////////////////////////////
// PROTECTED METHODS
//////////////////////////////////////////////////////////////

protected function addLoaderListeners( loader:URLLoader ):void
{
loader.addEventListener(Event.OPEN, onLoadStart);
loader.addEventListener(ProgressEvent.PROGRESS, onLoadProgress);
loader.addEventListener(Event.COMPLETE, onLoadComplete);
loader.addEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}
```

```
protected function removeLoaderListeners( loader:URLLoader ):void
{
loader.removeEventListener(Event.OPEN, onLoadStart);
loader.removeEventListener(ProgressEvent.PROGRESS, onLoadProgress);
loader.removeEventListener(Event.COMPLETE, onLoadComplete);
loader.removeEventListener(IOErrorEvent.IO_ERROR, onLoadIOError);
}

//////////////////////////////////////////////////////////
// GETTERS / SETTERS
//////////////////////////////////////////////////////////

public override function get progress():Number
{
if (!_loader)
{
return 0;
}

return _loader.bytesLoaded / _loader.bytesTotal;
}
}
}

-----------------------------------
File 428: igw\com\service\server\BinaryInputStream.as
package com.service.server
{
import com.enum.server.RequestEnum;

import flash.utils.ByteArray;

public class BinaryInputStream extends ByteArray
{
public var validToken:Boolean = true;
public var sequenceToken:int =
RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START;

private var stringInputCache:StringInputCache; // Current cache as set by setProtocol
private var stringInputCacheByProtocol:Array = new Array;
public var battleStringInputCache:StringInputCache = new StringInputCache;
public var sectorAlwaysVisibleStringInputCache:StringInputCache = new StringInputCache;
public var starbaseBaselineStringInputCache:StringInputCache = new StringInputCache;

public function BinaryInputStream()
{
super();
}

public
```

```
function setStringInputCache( stringCache:StringInputCache ):void
{
this.stringInputCache = stringCache;
}

public function checkToken():void
{
var token:int = this.readInt();
if (token != sequenceToken)
{
//validToken = false;
//++sequenceToken;
//return;
throw new Error("Bad BinaryInputStream sequence token, the calling read() function does not
match the Server data sent.");
}
++sequenceToken;
}

public function readStringCacheBaseline():void
{
if(!validToken)
return;

if (this.stringInputCache)
{
return this.stringInputCache.readBaseline(this);
}
}

public override function readUTF():String
{
if(!validToken)
return "";

if (this.stringInputCache)
{
return this.stringInputCache.readUTF(this);
}
return super.readUTF();
}

public override function readDouble():Number
{
if(!validToken)
return 0;

return super.readFloat();
}

public
```

```
override function clear():void
{
super.clear();
sequenceToken = RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START;
validToken = true;
this.stringInputCache = null;
}

public function readInt64():Number
{
if(!validToken)
return 0;

var high:int = this.readInt();
var low:uint = this.readUnsignedInt();
var retval:Number = Number(high) * Number(4294967296) + Number(low);
return retval;
}
}
}

------------------------------------
File 429: igw\com\service\server\EightTrack.as
package com.service.server
{
import com.Application;
import com.controller.ServerController;
import com.enum.server.EncodingEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.model.player.CurrentUser;
import com.service.server.outgoing.proxy.ProxyReportCrashRequest;

import flash.events.Event;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.events.SecurityErrorEvent;
import flash.utils.ByteArray;
import flash.utils.Endian;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;
import org.zlibfromscratch.ZlibDecoder;
import org.zlibfromscratch.ZlibDecoderError;

public class EightTrack
{
private const _logger:ILogger = getLogger("EightTrack");

private
```

```
var _io:BinaryInputStream;
private var _protocolListener:int = -1;
private var _responseSignal:Signal;
private var _serverController:ServerController;
private var _zlibDecoderBuffer:ByteArray;
private var _zlibDecoder:ZlibDecoder;
private var _waitingResponseSize:uint = 0;
private var _recordedStream:ByteArray;

public function init( stream:ByteArray ):void
{
_io = new BinaryInputStream();
_responseSignal = new Signal(IResponse);

_zlibDecoderBuffer = new ByteArray();
_zlibDecoderBuffer.endian = Endian.BIG_ENDIAN;
_zlibDecoder = new ZlibDecoder();

_recordedStream = stream;
_recordedStream.endian = Endian.BIG_ENDIAN;

reset();
}

public function reset():void
{
_recordedStream.position = 0;
_zlibDecoder.reset(false, false);
}

public function onReceive( e:ProgressEvent ):void
{
do
{
var size:uint;
if (_waitingResponseSize > 0)
{
size = _waitingResponseSize;
} else
{
if (_recordedStream.bytesAvailable >= 4)
size = _recordedStream.readUnsignedInt() - 4;
else
break;
}

if (_recordedStream.bytesAvailable >= size)
{
_io.clear();
_recordedStream.readBytes(_io, 0, size);
var
```

```
protocolID:int = _io.readByte();
var speakerID:int = _io.readByte();
var header:int = _io.readByte();
var encoding:int = _io.readByte();
if (encoding == EncodingEnum.BINARYZLIBCOMPRESSED)
{
// Read compressed input into _zlibDecoderBuffer.
_zlibDecoderBuffer.length = _io.bytesAvailable + 4;
_zlibDecoderBuffer.position = 0;
_io.readBytes(_zlibDecoderBuffer);

// Append the zlib sync-flush byte sequence.
_zlibDecoderBuffer.position = _zlibDecoderBuffer.length - 4;
_zlibDecoderBuffer.writeByte(0x00);
_zlibDecoderBuffer.writeByte(0x00);
_zlibDecoderBuffer.writeByte(0xff);
_zlibDecoderBuffer.writeByte(0xff);

// Uncompress the input to _io.
_zlibDecoderBuffer.position = 0;
_io.clear();
var bytesRead:uint = _zlibDecoder.feed(_zlibDecoderBuffer, _io);
if (bytesRead != _zlibDecoderBuffer.length || (_zlibDecoder.lastError !=
ZlibDecoderError.NEED_MORE_DATA && _zlibDecoder.lastError !=
ZlibDecoderError.NO_ERROR))
throw new Error("Failed to uncompress network message");

_io.position = 0;
encoding = EncodingEnum.BINARY;
}
if (_protocolListener == -1 || _protocolListener == protocolID ||
PacketFactory.isImportant(protocolID, header))
{
var response:IResponse = PacketFactory.getResponse(_io, protocolID, header, encoding);
if (response)
{
_responseSignal.dispatch(response);
} else
{
_logger.error("Unknown Message Received {0}, {1}, {2}", [protocolID, speakerID, header]);
var msg:ProxyReportCrashRequest =
ProxyReportCrashRequest(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_REPORT_CRASH));
msg.dataStr = 'Unknown Message Received ' + protocolID + ', ' + speakerID + ', ' + header;
_serverController.send(msg);
if (CONFIG::DEBUG)
throw new Error("Unknown Message Received");
}
}
_waitingResponseSize = 0;
}
```

```
else
{
_waitingResponseSize = size;
break;
}
} while (_recordedStream && _recordedStream.bytesAvailable > 0);
}

private function unknownMessage( header:int ):void
{
_logger.error("Unknown Message Received {0}, {1}, {2}", [header,
_recordedStream.bytesAvailable,
_recordedStream.readUTFBytes(_recordedStream.bytesAvailable)]);
}

private function postTo80Error( e:IOErrorEvent ):void
{
e.stopImmediatePropagation();
}

private function onError( e:IOErrorEvent ):void
{
_logger.error('IOError parsing data for replay');
}

public function addResponseListener( callback:Function ):void { _responseSignal.add(callback);
}
public function removeResponseListener( callback:Function ):void {
_responseSignal.remove(callback); }

public function get protocolListener():int { return _protocolListener; }
public function set protocolListener( v:int ):void { _protocolListener = v; }
public function set serverController( value:ServerController ):void { _serverController = value; }

public function destroy():void
{
_zlibDecoder.dispose();
_zlibDecoder = null;
_zlibDecoderBuffer.clear();
_zlibDecoderBuffer = null;

_io.clear();
_io = null;
_responseSignal.removeAll();
_responseSignal = null;
_serverController = null;
_recordedStream = null;
}

}
}
```

```
------------------------------------
File 430: igw\com\service\server\IRequest.as
package com.service.server
{
import flash.utils.ByteArray;

public interface IRequest
{
function init( protocolID:int, header:int ):void;

function write( output:ByteArray ):void;

function destroy():void;
}

}

------------------------------------
File 431: igw\com\service\server\IResponse.as
package com.service.server
{
import flash.utils.ByteArray;

public interface IResponse
{
function read( input:BinaryInputStream ):void;
function readJSON( input:Object ):void;

function get isTicked():Boolean;

function get header():int;
function set header( v:int ):void;

function get protocolID():int;
function set protocolID( v:int ):void;

function destroy():void;
}

}

------------------------------------
File 432: igw\com\service\server\ITickedResponse.as
package com.service.server
{
public interface ITickedResponse extends IResponse
{
function get isBaseline():Boolean;

function
```

```
get addTick():Boolean;

function get tick():int;

function get timeStep():int;
}
}

------------------------------------
File 433: igw\com\service\server\ITransactionRequest.as
package com.service.server
{
public interface ITransactionRequest extends IRequest
{
function get token():int;
function set token( v:int ):void;
}
}

------------------------------------
File 434: igw\com\service\server\ITransactionResponse.as
package com.service.server
{
import com.model.transaction.TransactionVO;

public interface ITransactionResponse extends IResponse
{
function get data():TransactionVO;
function set data( v:TransactionVO ):void;
function get success():Boolean
function get token():int;
}
}

------------------------------------
File 435: igw\com\service\server\PacketFactory.as
package com.service.server
{
import com.enum.server.EncodingEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.enum.server.ResponseEnum;
import com.service.server.incoming.alliance.AllianceBaselineResponse;
import com.service.server.incoming.alliance.AllianceGenericResponse;
import com.service.server.incoming.alliance.AllianceInviteResponse;
import com.service.server.incoming.alliance.AllianceRosterResponse;
import com.service.server.incoming.alliance.PublicAlliancesResponse;
import com.service.server.incoming.battle.BattleDataResponse;
import com.service.server.incoming.battle.BattleDebugLinesResponse;
import com.service.server.incoming.battle.BattleHasEndedResponse;
import
```

com.service.server.incoming.battlelog.BattleLogDetailsResponse;
import com.service.server.incoming.battlelog.BattleLogListResponse;
import com.service.server.incoming.chat.ChatBaselineResponse;
import com.service.server.incoming.chat.ChatEventResponse;
import com.service.server.incoming.chat.ChatResponse;
import com.service.server.incoming.leaderboard.LeaderboardResponse;
import com.service.server.incoming.leaderboard.PlayerProfileResponse;
import com.service.server.incoming.leaderboard.WarfrontUpdateResponse;
import com.service.server.incoming.mail.MailDetailResponse;
import com.service.server.incoming.mail.MailInboxResponse;
import com.service.server.incoming.mail.MailUnreadResponse;
import com.service.server.incoming.proxy.ProxyBattleDisconnectedResponse;
import com.service.server.incoming.proxy.ProxySectorDisconnectedResponse;
import com.service.server.incoming.proxy.ProxyStarbaseDisconnectedResponse;
import com.service.server.incoming.proxy.TimeSyncResponse;
import com.service.server.incoming.sector.SectorAlwaysVisibleBaselineResponse;
import com.service.server.incoming.sector.SectorAlwaysVisibleUpdateResponse;
import com.service.server.incoming.sector.SectorBaselineResponse;
import com.service.server.incoming.sector.SectorFleetTravelAlertResponse;
import com.service.server.incoming.universe.UniverseSectorListResponse;
import com.service.server.incoming.sector.SectorUpdateResponse;
import com.service.server.incoming.starbase.StarbaseAchievementsResponse;
import com.service.server.incoming.starbase.StarbaseAllScoresResponse;
import com.service.server.incoming.starbase.StarbaseAvailableRerollResponse;
import com.service.server.incoming.starbase.StarbaseBaselineResponse;
import com.service.server.incoming.starbase.StarbaseBattleAlertResponse;
import com.service.server.incoming.starbase.StarbaseBountyRewardResponse;
import com.service.server.incoming.starbase.StarbaseDailyResponse;
import com.service.server.incoming.starbase.StarbaseDailyRewardResponse;
import com.service.server.incoming.starbase.StarbaseFleetDockedResponse;
import com.service.server.incoming.starbase.StarbaseGetPaywallPayoutsResponse;
import com.service.server.incoming.starbase.StarbaseInstancedMissionAlertResponse;
import com.service.server.incoming.starbase.StarbaseMissionCompleteResponse;
import com.service.server.incoming.starbase.StarbaseMotdListResponse;
import com.service.server.incoming.starbase.StarbaseMoveStarbaseResponse;
import com.service.server.incoming.starbase.StarbaseOfferRedeemedResponse;
import com.service.server.incoming.starbase.StarbaseRerollChanceResultResponse;
import com.service.server.incoming.starbase.StarbaseRerollReceivedResultResponse;
import com.service.server.incoming.starbase.StarbaseTransactionResponse;
import com.service.server.incoming.starbase.StarbaseUnavailableRerollResponse;
import com.service.server.incoming.universe.UniverseNeedCharacterCreateResponse;
import com.service.server.outgoing.alliance.AllianceCreateAllianceRequest;
import com.service.server.outgoing.alliance.AllianceDemoteRequest;
import com.service.server.outgoing.alliance.AllianceIgnoreInvitesRequest;
import com.service.server.outgoing.alliance.AllianceJoinRequest;
import com.service.server.outgoing.alliance.AllianceKickRequest;
import com.service.server.outgoing.alliance.AllianceLeaveRequest;
import com.service.server.outgoing.alliance.AlliancePromoteRequest;
import com.service.server.outgoing.alliance.AllianceRequestBaselineRequest;
import com.service.server.outgoing.alliance.AllianceRequestPublicsRequest;
import

com.service.server.outgoing.alliance.AllianceRequestRosterRequest;
import com.service.server.outgoing.alliance.AllianceSendInviteRequest;
import com.service.server.outgoing.alliance.AllianceSetDescriptionRequest;
import com.service.server.outgoing.alliance.AllianceSetMOTDRequest;
import com.service.server.outgoing.alliance.AllianceSetPublicRequest;
import com.service.server.outgoing.battle.BattleAttackOrderRequest;
import com.service.server.outgoing.battle.BattleMoveOrderRequest;
import com.service.server.outgoing.battle.BattlePauseRequest;
import com.service.server.outgoing.battle.BattleRetreatRequest;
import com.service.server.outgoing.battle.BattleToggleModuleOrderRequest;
import com.service.server.outgoing.battlelog.BattleLogDetailRequest;
import com.service.server.outgoing.battlelog.BattleLogListRequest;
import com.service.server.outgoing.chat.ChatChangeRoomRequest;
import com.service.server.outgoing.chat.ChatIgnoreChatRequest;
import com.service.server.outgoing.chat.ChatReportChatRequest;
import com.service.server.outgoing.chat.ChatSendChatRequest;
import com.service.server.outgoing.leaderboard.LeaderboardRequest;
import com.service.server.outgoing.leaderboard.LeaderboardRequestPlayerProfileRequest;
import com.service.server.outgoing.mail.MailDeleteMailRequest;
import com.service.server.outgoing.mail.MailReadMailRequest;
import com.service.server.outgoing.mail.MailRequestInboxRequest;
import com.service.server.outgoing.mail.MailSendAllianceMailRequest;
import com.service.server.outgoing.mail.MailSendMailRequest;
import com.service.server.outgoing.proxy.AuthRequest;
import com.service.server.outgoing.proxy.ClientLoginRequest;
import com.service.server.outgoing.proxy.ProxyConnectToBattleRequest;
import com.service.server.outgoing.proxy.ProxyConnectToSectorRequest;
import com.service.server.outgoing.proxy.ProxyReportCrashRequest;
import com.service.server.outgoing.proxy.ProxyReportLoginDataRequest;
import com.service.server.outgoing.proxy.ProxyTutorialStepCompletedMessage;
import com.service.server.outgoing.proxy.TimeSyncRequest;
import com.service.server.outgoing.sector.SectorOrderRequest;
import com.service.server.outgoing.sector.SectorRequestBaselineRequest;
import com.service.server.outgoing.sector.SectorSetViewLocationRequest;
import com.service.server.outgoing.starbase.StarbaseBookmarkDeleteRequest;
import com.service.server.outgoing.starbase.StarbaseBookmarkSaveRequest;
import com.service.server.outgoing.starbase.StarbaseBookmarkUpdateRequest;
import com.service.server.outgoing.starbase.StarbaseBribeContractRequest;
import com.service.server.outgoing.starbase.StarbaseBuildNewBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseBuildShipRequest;
import com.service.server.outgoing.starbase.StarbaseBuyResourceRequest;
import com.service.server.outgoing.starbase.StarbaseBuyStoreItemRequest;
import com.service.server.outgoing.starbase.StarbaseBuyOtherStoreItemRequest;
import com.service.server.outgoing.starbase.StarbaseBuyoutBlueprintRequest;
import com.service.server.outgoing.starbase.StarbaseCancelContractRequest;
import com.service.server.outgoing.starbase.StarbaseCancelTransactionRequest;
import com.service.server.outgoing.starbase.StarbaseClaimAchievementRewardRequest;
import com.service.server.outgoing.starbase.StarbaseMintNFTRequest;
import com.service.server.outgoing.starbase.StarbaseClaimDailyRequest;
import com.service.server.outgoing.starbase.StarbaseExtendContractRequest;
import

```
com.service.server.outgoing.starbase.StarbaseGetPaywallPayoutsRequest;
import com.service.server.outgoing.starbase.StarbaseInstancedMissionStartRequest;
import com.service.server.outgoing.starbase.StarbaseLaunchFleetRequest;
import com.service.server.outgoing.starbase.StarbaseMissionAcceptRequest;
import com.service.server.outgoing.starbase.StarbaseMissionAcceptRewardsRequest;
import com.service.server.outgoing.starbase.StarbaseCompleteBlueprintResearchRequest;
import com.service.server.outgoing.starbase.StarbaseMissionStepRequest;
import com.service.server.outgoing.starbase.StarbaseMotDReadRequest;
import com.service.server.outgoing.starbase.StarbaseMoveBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseMoveStarbaseRequest;
import com.service.server.outgoing.starbase.StarbaseMoveStarbaseToTransgateRequest;

import com.service.server.outgoing.starbase.StarbaseNegotiateContractRequest;
import com.service.server.outgoing.starbase.StarbaseRecallFleetRequest;
import com.service.server.outgoing.starbase.StarbaseRecycleBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseRecycleShipRequest;
import com.service.server.outgoing.starbase.StarbaseRefitBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseRefitShipRequest;
import com.service.server.outgoing.starbase.StarbaseRenameFleetRequest;
import com.service.server.outgoing.starbase.StarbaseRenamePlayerRequest;
import com.service.server.outgoing.starbase.StarbaseRepairBaseRequest;
import com.service.server.outgoing.starbase.StarbaseRepairFleetRequest;
import com.service.server.outgoing.starbase.StarbaseAllScoresRequest;
import com.service.server.outgoing.starbase.StarbaseRequestAchievementsRequest;
import com.service.server.outgoing.starbase.StarbaseRerollBlueprintChanceRequest;
import com.service.server.outgoing.starbase.StarbaseRerollBlueprintReceivedRequest;
import com.service.server.outgoing.starbase.StarbaseResearchRequest;
import com.service.server.outgoing.starbase.StarbaseResecureContractRequest;
import com.service.server.outgoing.starbase.StarbaseSetClientSettingsRequest;
import com.service.server.outgoing.starbase.StarbaseSkipTrainingRequest;
import com.service.server.outgoing.starbase.StarbaseSpeedUpTransactionRequest;
import com.service.server.outgoing.starbase.StarbaseUpdateFleetRequest;
import com.service.server.outgoing.starbase.StarbaseUpgradeBuildingRequest;
import com.service.server.outgoing.starbase.StarbaseVerifyPaymentRequest;
import com.service.server.outgoing.universe.UniverseCreateCharacterRequest;

import org.shared.ObjectPool;

public class PacketFactory
{
public static function getResponse( input:BinaryInputStream, protocolID:int, header:int,
encoding:int ):IResponse
{
var packet:IResponse = null;
switch (protocolID)
{
case ProtocolEnum.PROXY_CLIENT:
switch (header)
{
case ResponseEnum.PROXY_TIME_SYNC:
packet
```

```
= ObjectPool.get(TimeSyncResponse);
break;
case ResponseEnum.PROXY_BATTLE_DISCONNECTED:
packet = ObjectPool.get(ProxyBattleDisconnectedResponse);
break;
case ResponseEnum.PROXY_SECTOR_DISCONNECTED:
packet = ObjectPool.get(ProxySectorDisconnectedResponse);
break;
case ResponseEnum.PROXY_STARBASE_DISCONNECTED:
packet = ObjectPool.get(ProxyStarbaseDisconnectedResponse);
break;
}
break;
case ProtocolEnum.SECTOR_CLIENT:
switch (header)
{
case ResponseEnum.SECTOR_BASELINE:
packet = ObjectPool.get(SectorBaselineResponse);
break;
case ResponseEnum.SECTOR_UPDATE:
packet = ObjectPool.get(SectorUpdateResponse);
break;
case ResponseEnum.SECTOR_ALWAYS_VISIBLE_BASELINE:
packet = ObjectPool.get(SectorAlwaysVisibleBaselineResponse);
break;
case ResponseEnum.SECTOR_ALWAYS_VISIBLE_UPDATE:
packet = ObjectPool.get(SectorAlwaysVisibleUpdateResponse);
break;
case ResponseEnum.SECTOR_FLEET_TRAVEL_ALERT:
packet = ObjectPool.get(SectorFleetTravelAlertResponse);
break;
}
break;
case ProtocolEnum.BATTLE_CLIENT:
switch (header)
{
case ResponseEnum.BATTLE_BASELINE:
case ResponseEnum.BATTLE_UPDATE:
packet = ObjectPool.get(BattleDataResponse);
break;
case ResponseEnum.BATTLE_DEBUG_LINES:
packet = ObjectPool.get(BattleDebugLinesResponse);
break;
case ResponseEnum.BATTLE_HAS_ENDED:
packet = ObjectPool.get(BattleHasEndedResponse);
break;
}
break;
case ProtocolEnum.STARBASE_CLIENT:
switch (header)
{
```

```
case ResponseEnum.STARBASE_TRANSACTION_RESPONSE:
packet = ObjectPool.get(StarbaseTransactionResponse);
break;
case ResponseEnum.STARBASE_BASELINE:
packet = ObjectPool.get(StarbaseBaselineResponse);
break;
case ResponseEnum.STARBASE_BATTLE_ALERT:
packet = ObjectPool.get(StarbaseBattleAlertResponse);
break;
case ResponseEnum.STARBASE_MISSION_COMPLETE:
packet = ObjectPool.get(StarbaseMissionCompleteResponse);
break;
case ResponseEnum.STARBASE_FLEET_DOCKED:
packet = ObjectPool.get(StarbaseFleetDockedResponse);
break;
case ResponseEnum.STARBASE_BOUNTY_REWARD:
packet = ObjectPool.get(StarbaseBountyRewardResponse);
break;
case ResponseEnum.STARBASE_BATTLELOG_LIST:
packet = ObjectPool.get(BattleLogListResponse);
break;
case ResponseEnum.STARBASE_BATTLELOG_DETAILS:
packet = ObjectPool.get(BattleLogDetailsResponse);
break;
case ResponseEnum.STARBASE_OFFER_REDEEMED:
packet = ObjectPool.get(StarbaseOfferRedeemedResponse);
break;
case ResponseEnum.STARBASE_MOTD_LIST:
packet = ObjectPool.get(StarbaseMotdListResponse);
break;
case ResponseEnum.STARBASE_DAILY:
packet = ObjectPool.get(StarbaseDailyResponse);
break;
case ResponseEnum.STARBASE_DAILY_REWARD:
packet = ObjectPool.get(StarbaseDailyRewardResponse);
break;
case ResponseEnum.STARBASE_AVAILABLE_REROLL:
case ResponseEnum.STARBASE_AVAILABLE_CREWMEMBER_REROLL:
packet = ObjectPool.get(StarbaseAvailableRerollResponse);
break;
case ResponseEnum.STARBASE_REROLL_CHANCE_RESULT:
packet = ObjectPool.get(StarbaseRerollChanceResultResponse);
break;
case ResponseEnum.STARBASE_REROLL_RECEIVED_RESULT:
case ResponseEnum.STARBASE_REROLL_CREWMEMBER_RECEIVED_RESULT:
packet = ObjectPool.get(StarbaseRerollReceivedResultResponse);
break;
case ResponseEnum.STARBASE_MOVE_STARBASE_RESPONSE:
packet = ObjectPool.get(StarbaseMoveStarbaseResponse);
break;
```

```
case ResponseEnum.STARBASE_ACHIEVEMENTS_RESPONSE:
packet = ObjectPool.get(StarbaseAchievementsResponse);
break;
case ResponseEnum.STARBASE_ALL_SCORES_RESPONSE:
packet = ObjectPool.get(StarbaseAllScoresResponse);
break;
case ResponseEnum.STARBASE_UNAVAILABLE_REROLL:
case ResponseEnum.STARBASE_UNAVAILABLE_CREWMEMBER_REROLL:
packet = ObjectPool.get(StarbaseUnavailableRerollResponse);
break;
case ResponseEnum.STARBASE_GET_PAYWALL_PAYOUTS_RESPONSE:
packet = ObjectPool.get(StarbaseGetPaywallPayoutsResponse);
break;
case ResponseEnum.STARBASE_INSTANCED_MISSION_ALERT:
packet = ObjectPool.get(StarbaseInstancedMissionAlertResponse);
break;
}
break;
case ProtocolEnum.MAIL_CLIENT:
switch (header)
{
case ResponseEnum.MAIL_UNREAD:
packet = ObjectPool.get(MailUnreadResponse);
break;
case ResponseEnum.MAIL_INBOX:
packet = ObjectPool.get(MailInboxResponse);
break;
case ResponseEnum.MAIL_DETAIL:
packet = ObjectPool.get(MailDetailResponse);
break;
}
break;
case ProtocolEnum.ALLIANCE_CLIENT:
switch (header)
{
case ResponseEnum.ALLIANCE_BASELINE:
packet = ObjectPool.get(AllianceBaselineResponse);
break;
case ResponseEnum.ALLIANCE_ROSTER:
packet = ObjectPool.get(AllianceRosterResponse);
break;
case ResponseEnum.GENERIC_ALLIANCE_RESPONSE:
packet = ObjectPool.get(AllianceGenericResponse);
break;
case ResponseEnum.PUBLIC_ALLIANCES_RESPONSE:
packet = ObjectPool.get(PublicAlliancesResponse);
break;
case ResponseEnum.ALLIANCE_INVITE:
packet = ObjectPool.get(AllianceInviteResponse);
break;
```

```
}
break;
case ProtocolEnum.CHAT_CLIENT:
switch (header)
{
case ResponseEnum.CHAT_RESPONSE:
packet = ObjectPool.get(ChatResponse);
break;
case ResponseEnum.CHAT_BASELINE:
packet = ObjectPool.get(ChatBaselineResponse);
break;
case ResponseEnum.CHAT_EVENT:
packet = ObjectPool.get(ChatEventResponse);
break;
}
break;
case ProtocolEnum.LEADERBOARD_CLIENT:
switch (header)
{
case ResponseEnum.LEADERBOARD:
packet = ObjectPool.get(LeaderboardResponse);
break;
case ResponseEnum.PLAYER_PROFILE:
packet = ObjectPool.get(PlayerProfileResponse);
break;
case ResponseEnum.WARFRONT_UPDATE:
packet = ObjectPool.get(WarfrontUpdateResponse);
break;
}
break;

case ProtocolEnum.UNIVERSE_CLIENT:
switch (header)
{
case ResponseEnum.UNIVERSE_NEED_CHARACTER_CREATE:
packet = ObjectPool.get(UniverseNeedCharacterCreateResponse);
break;
case ResponseEnum.UNIVERSE_SECTOR_LIST:
packet = ObjectPool.get(UniverseSectorListResponse);
break;
}
break;
}

if (packet)
{
packet.header = header;
packet.protocolID = protocolID;
switch (encoding)
{
```

```
case EncodingEnum.BINARY:
packet.read(input);
break;
case EncodingEnum.JSON:
var jsonString:String = String(input.readUTFBytes(input.length - 4));
var data:Object = JSON.parse(jsonString);
packet.readJSON(data);
break;
}
}
return packet;
}

public static function getRequest( protocolID:int, header:int ):IRequest
{
var packet:IRequest = null;
switch (protocolID)
{
case ProtocolEnum.PROXY_CLIENT:
switch (header)
{
case RequestEnum.AUTHORIZATION:
packet = ObjectPool.get(AuthRequest);
break;
case RequestEnum.PROXY_TIME_SYNC:
packet = ObjectPool.get(TimeSyncRequest);
break;
case RequestEnum.PROXY_CONNECT_TO_BATTLE:
packet = ObjectPool.get(ProxyConnectToBattleRequest);
break;
case RequestEnum.PROXY_CONNECT_TO_SECTOR:
packet = ObjectPool.get(ProxyConnectToSectorRequest);
break;
case RequestEnum.PROXY_LOGIN:
packet = ObjectPool.get(ClientLoginRequest);
break;
case RequestEnum.PROXY_REPORT_CRASH:
packet = ObjectPool.get(ProxyReportCrashRequest);
break;
case RequestEnum.PROXY_REPORT_LOGIN_DATA:
packet = ObjectPool.get(ProxyReportLoginDataRequest);
break;
case RequestEnum.PROXY_TUTORIAL_STEP_COMPLETED:
packet = ObjectPool.get(ProxyTutorialStepCompletedMessage);
break;
}
break;
case ProtocolEnum.SECTOR_CLIENT:
switch (header)
{
```

```
case RequestEnum.SECTOR_SET_VIEW_LOCATION:
packet = ObjectPool.get(SectorSetViewLocationRequest);
break;
case RequestEnum.SECTOR_ISSUE_ORDER:
packet = ObjectPool.get(SectorOrderRequest);
break;
case RequestEnum.SECTOR_REQUEST_BASELINE:
packet = ObjectPool.get(SectorRequestBaselineRequest);
break;
}
break;
case ProtocolEnum.BATTLE_CLIENT:
switch (header)
{
case RequestEnum.BATTLE_MOVE_ORDER:
packet = ObjectPool.get(BattleMoveOrderRequest);
break;
case RequestEnum.BATTLE_ATTACK_ORDER:
packet = ObjectPool.get(BattleAttackOrderRequest);
break;
case RequestEnum.BATTLE_TOGGLE_MODULE_ORDER:
packet = ObjectPool.get(BattleToggleModuleOrderRequest);
break;
case RequestEnum.BATTLE_PAUSE:
packet = ObjectPool.get(BattlePauseRequest);
break;
case RequestEnum.BATTLE_RETREAT:
packet = ObjectPool.get(BattleRetreatRequest);
break;
}
break;
case ProtocolEnum.STARBASE_CLIENT:
switch (header)
{
case RequestEnum.STARBASE_BUILD_SHIP:
packet = ObjectPool.get(StarbaseBuildShipRequest);
break;
case RequestEnum.STARBASE_UPDATE_FLEET:
packet = ObjectPool.get(StarbaseUpdateFleetRequest);
break;
case RequestEnum.STARBASE_LAUNCH_FLEET:
packet = ObjectPool.get(StarbaseLaunchFleetRequest);
break;
case RequestEnum.STARBASE_RECALL_FLEET:
packet = ObjectPool.get(StarbaseRecallFleetRequest);
break;
case RequestEnum.STARBASE_REPAIR_FLEET:
packet = ObjectPool.get(StarbaseRepairFleetRequest);
break;
case
```

```
RequestEnum.STARBASE_RENAME_FLEET:
packet = ObjectPool.get(StarbaseRenameFleetRequest);
break;
case RequestEnum.STARBASE_BUILD_NEW_BUILDING:
packet = ObjectPool.get(StarbaseBuildNewBuildingRequest);
break;
case RequestEnum.STARBASE_SET_CLIENT_SETTINGS:
packet = ObjectPool.get(StarbaseSetClientSettingsRequest);
break;
case RequestEnum.STARBASE_UPGRADE_BUILDING:
packet = ObjectPool.get(StarbaseUpgradeBuildingRequest);
break;
case RequestEnum.STARBASE_RECYCLE_BUILDING:
packet = ObjectPool.get(StarbaseRecycleBuildingRequest);
break;
case RequestEnum.STARBASE_REFIT_BUILDING:
packet = ObjectPool.get(StarbaseRefitBuildingRequest);
break;
case RequestEnum.STARBASE_REPAIR_BASE:
packet = ObjectPool.get(StarbaseRepairBaseRequest);
break;
case RequestEnum.STARBASE_SPEED_UP_TRANSACTION:
packet = ObjectPool.get(StarbaseSpeedUpTransactionRequest);
break;
case RequestEnum.STARBASE_CANCEL_TRANSACTION:
packet = ObjectPool.get(StarbaseCancelTransactionRequest);
break;
case RequestEnum.STARBASE_MOVE_BUILDING:
packet = ObjectPool.get(StarbaseMoveBuildingRequest);
break;
case RequestEnum.STARBASE_RESEARCH:
packet = ObjectPool.get(StarbaseResearchRequest);
break;
case RequestEnum.STARBASE_BUY_RESOURCE:
packet = ObjectPool.get(StarbaseBuyResourceRequest);
break;
case RequestEnum.STARBASE_BUY_STORE_ITEM:
packet = ObjectPool.get(StarbaseBuyStoreItemRequest);
break;
case RequestEnum.STARBASE_BUY_OTHER_STORE_ITEM:
packet = ObjectPool.get(StarbaseBuyOtherStoreItemRequest);
break;
case RequestEnum.STARBASE_RECYCLE_SHIP:
packet = ObjectPool.get(StarbaseRecycleShipRequest);
break;
case RequestEnum.STARBASE_REFIT_SHIP:
packet = ObjectPool.get(StarbaseRefitShipRequest);
break;
case RequestEnum.STARBASE_NEGOTIATE_CONTRACT:
packet = ObjectPool.get(StarbaseNegotiateContractRequest);
break;
```

```
case RequestEnum.STARBASE_BRIBE_CONTRACT:
packet = ObjectPool.get(StarbaseBribeContractRequest);
break;
case RequestEnum.STARBASE_CANCEL_CONTRACT:
packet = ObjectPool.get(StarbaseCancelContractRequest);
break;
case RequestEnum.STARBASE_EXTEND_CONTRACT:
packet = ObjectPool.get(StarbaseExtendContractRequest);
break;
case RequestEnum.STARBASE_RESECURE_CONTRACT:
packet = ObjectPool.get(StarbaseResecureContractRequest);
break;
case RequestEnum.STARBASE_INSTANCED_MISSION_START:
packet = ObjectPool.get(StarbaseInstancedMissionStartRequest);
break;
case RequestEnum.STARBASE_MISSION_STEP:
packet = ObjectPool.get(StarbaseMissionStepRequest);
break;
case RequestEnum.STARBASE_MISSION_ACCEPT:
packet = ObjectPool.get(StarbaseMissionAcceptRequest);
break;
case RequestEnum.STARBASE_MISSION_ACCEPT_REWARDS:
packet = ObjectPool.get(StarbaseMissionAcceptRewardsRequest);
break;
case RequestEnum.STARBASE_BUYOUT_BLUEPRINT:
packet = ObjectPool.get(StarbaseBuyoutBlueprintRequest);
break;
case RequestEnum.STARBASE_COMPLETE_BLUEPRINT_RESEARCH:
packet = ObjectPool.get(StarbaseCompleteBlueprintResearchRequest);
break;
case RequestEnum.STARBASE_BATTLELOG_LIST:
packet = ObjectPool.get(BattleLogListRequest);
break;
case RequestEnum.STARBASE_BATTLELOG_DETAILS:
packet = ObjectPool.get(BattleLogDetailRequest);
break;
case RequestEnum.STARBASE_BOOKMARK_SAVE:
packet = ObjectPool.get(StarbaseBookmarkSaveRequest);
break;
case RequestEnum.STARBASE_BOOKMARK_DELETE:
packet = ObjectPool.get(StarbaseBookmarkDeleteRequest);
break;
case RequestEnum.STARBASE_BOOKMARK_UPDATE:
packet = ObjectPool.get(StarbaseBookmarkUpdateRequest);
break;
case RequestEnum.STARBASE_MARK_MOTD_READ_MESSAGE:
packet = ObjectPool.get(StarbaseMotDReadRequest);
break;
case RequestEnum.STARBASE_CLAIM_DAILY_MESSAGE:
packet
```

```
= ObjectPool.get(StarbaseClaimDailyRequest);
break;
case RequestEnum.STARBASE_SKIP_TRAINING_MESSAGE:
packet = ObjectPool.get(StarbaseSkipTrainingRequest);
break;
case RequestEnum.STARBASE_REROLL_BLUEPRINT_CHANCE_MESSAGE:
packet = ObjectPool.get(StarbaseRerollBlueprintChanceRequest);
break;
case RequestEnum.STARBASE_REROLL_BLUEPRINT_RECEIVED_MESSAGE:
packet = ObjectPool.get(StarbaseRerollBlueprintReceivedRequest);
break;
case RequestEnum.STARBASE_RENAME_PLAYER:
packet = ObjectPool.get(StarbaseRenamePlayerRequest);
break;
case RequestEnum.STARBASE_MOVE_STARBASE:
packet = ObjectPool.get(StarbaseMoveStarbaseRequest);
break;
case RequestEnum.STARBASE_MOVE_STARBASE_TO_TRANSGATE:
packet = ObjectPool.get(StarbaseMoveStarbaseToTransgateRequest);
break;
case RequestEnum.STARBASE_REQUEST_ACHIEVEMENTS:
packet = ObjectPool.get(StarbaseRequestAchievementsRequest);
break;
case RequestEnum.STARBASE_REQUEST_ALL_SCORES:
packet = ObjectPool.get(StarbaseAllScoresRequest);
break;
case RequestEnum.STARBASE_CLAIM_ACHIEVEMENT_REWARD:
packet = ObjectPool.get(StarbaseClaimAchievementRewardRequest);
break;
case RequestEnum.STARBASE_GET_PAYWALL_PAYOUTS:
packet = ObjectPool.get(StarbaseGetPaywallPayoutsRequest);
break;
case RequestEnum.STARBASE_VERIFY_PAYMENT:
packet = ObjectPool.get(StarbaseVerifyPaymentRequest);
break;
case RequestEnum.STARBASE_MINT_NFT:
packet = ObjectPool.get(StarbaseMintNFTRequest);
break;
}
break;

case ProtocolEnum.MAIL_CLIENT:
switch (header)
{
case RequestEnum.MAIL_REQUEST_INBOX:
packet = ObjectPool.get(MailRequestInboxRequest);
break;
case RequestEnum.MAIL_SEND_MAIL:
packet = ObjectPool.get(MailSendMailRequest);
break;
case
```

```
RequestEnum.MAIL_DELETE_MAIL:
packet = ObjectPool.get(MailDeleteMailRequest);
break;
case RequestEnum.MAIL_READ_MAIL:
packet = ObjectPool.get(MailReadMailRequest);
break;
case RequestEnum.MAIL_SEND_ALLIANCE_MAIL:
packet = ObjectPool.get(MailSendAllianceMailRequest);
break;
}
break;

case ProtocolEnum.ALLIANCE_CLIENT:
switch (header)
{
case RequestEnum.ALLIANCE_REQUEST_BASELINE:
packet = ObjectPool.get(AllianceRequestBaselineRequest);
break;
case RequestEnum.ALLIANCE_REQUEST_ROSTER:
packet = ObjectPool.get(AllianceRequestRosterRequest);
break;
case RequestEnum.ALLIANCE_CREATE_ALLIANCE:
packet = ObjectPool.get(AllianceCreateAllianceRequest);
break;
case RequestEnum.ALLIANCE_SET_MOTD:
packet = ObjectPool.get(AllianceSetMOTDRequest);
break;
case RequestEnum.ALLIANCE_SET_DESCRIPTION:
packet = ObjectPool.get(AllianceSetDescriptionRequest);
break;
case RequestEnum.ALLIANCE_SET_PUBLIC:
packet = ObjectPool.get(AllianceSetPublicRequest);
break;
case RequestEnum.ALLIANCE_PROMOTE:
packet = ObjectPool.get(AlliancePromoteRequest);
break;
case RequestEnum.ALLIANCE_DEMOTE:
packet = ObjectPool.get(AllianceDemoteRequest);
break;
case RequestEnum.ALLIANCE_KICK:
packet = ObjectPool.get(AllianceKickRequest);
break;
case RequestEnum.ALLIANCE_LEAVE_ALLIANCE:
packet = ObjectPool.get(AllianceLeaveRequest);
break;
case RequestEnum.ALLIANCE_JOIN_ALLIANCE:
packet = ObjectPool.get(AllianceJoinRequest);
break;
case RequestEnum.ALLIANCE_SEND_INVITE:
packet = ObjectPool.get(AllianceSendInviteRequest);
break;
```

```
case RequestEnum.ALLIANCE_IGNORE_INVITES:
packet = ObjectPool.get(AllianceIgnoreInvitesRequest);
break;
case RequestEnum.ALLIANCE_REQUEST_PUBLICS:
packet = ObjectPool.get(AllianceRequestPublicsRequest);
break;
}
break;

case ProtocolEnum.CHAT_CLIENT:
switch (header)
{
case RequestEnum.CHAT_SEND_CHAT:
packet = ObjectPool.get(ChatSendChatRequest);
break;
case RequestEnum.CHAT_IGNORE_CHAT:
packet = ObjectPool.get(ChatIgnoreChatRequest);
break;
case RequestEnum.CHAT_REPORT_CHAT:
packet = ObjectPool.get(ChatReportChatRequest);
break;
case RequestEnum.CHAT_CHANGE_ROOM:
packet = ObjectPool.get(ChatChangeRoomRequest);
break;
}
break;

case ProtocolEnum.LEADERBOARD_CLIENT:
switch (header)
{
case RequestEnum.LEADERBOARD_REQUEST_LEADERBOARD:
packet = ObjectPool.get(LeaderboardRequest);
break;
case RequestEnum.LEADERBOARD_REQUEST_PLAYER_PROFILE:
packet = ObjectPool.get(LeaderboardRequestPlayerProfileRequest);
break;
}
break;

case ProtocolEnum.UNIVERSE_CLIENT:
switch (header)
{
case RequestEnum.UNIVERSE_CHARACTER_CREATION_REQUEST:
packet = ObjectPool.get(UniverseCreateCharacterRequest);
break;
}
break;
}
if (packet)
packet.init(protocolID,
```

```
header);
return packet;
}

public static function isImportant( protocolID:int, header:int ):Boolean
{
if (protocolID == ProtocolEnum.PROXY_CLIENT)
return true;
if (header == 1)
return true;
if (protocolID > 2)
return true;
return false;
}
}

}

-------------------------------------
File 436: igw\com\service\server\StringInputCache.as
package com.service.server
{
public class StringInputCache
{
private var _count:uint = 0;
private var _capacity:uint = 0;
private var _table:Vector.<String> = new Vector.<String>;

public function readBaseline( input:BinaryInputStream ):void
{
_table.length = 0;
input.checkToken();
_count = input.readUnsignedInt();
_capacity = input.readUnsignedShort();
var tableSize:int = Math.min(_count, _capacity);
for (var i:int = 0; i < tableSize; ++i)
{
var len:uint = input.readUnsignedShort();
_table.push(input.readUTFBytes(len));
}
input.checkToken();
}

public function readUTF( input:BinaryInputStream ):String
{
var id:uint = input.readUnsignedShort();
if (id >= _capacity)
{
// new ( or replacement ) entry
var len:uint = id - _capacity;
var
```

```
value:String = input.readUTFBytes(len);
if (_count < _capacity)
{
_table.push(value);
} else
{
id = (_count * 33331) % _capacity;
_table[id] = value;
}
++_count;

return value;
} else
{
if(id < _table.length)
{
return _table[id];
}
else
{
return "";
}
}
}
}
}
```

File 437: igw\com\service\server\TinCan.as

```
package com.service.server
{
import com.Application;
import com.controller.ServerController;
import com.enum.server.EncodingEnum;
import com.enum.server.ProtocolEnum;
import com.enum.server.RequestEnum;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.server.outgoing.proxy.ProxyReportCrashRequest;

import flash.events.Event;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.events.SecurityErrorEvent;
import flash.net.Socket;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;
import
```

```
flash.utils.ByteArray;
import flash.utils.Endian;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;
import org.zlibfromscratch.ZlibDecoder;
import org.zlibfromscratch.ZlibDecoderError;

public class TinCan
{
public static const CONNECTED:int = 0;
public static const CONNECTION_LOST:int = 1;
public static const CONNECTION_FAILED:int = 2;
public static const GAME:int = 0;
public static const CHAT:int = 1;

private const _logger:ILogger = getLogger("TinCan");
private var _connectionSignal:Signal;
private var _i:BinaryInputStream;
private var _o:BinaryInputStream;
private var _ip:String;
private var _port:int;
private var _devConnection:Boolean;
private var _fullSize:uint = 0;
private var _simRead:int = 0;
private var _simWrite:int = 0;
private var _lockRead:int = 0;
private var _protocolListener:int = -1;
private var _responseSignal:Signal;
private var _serverController:ServerController;
private var _zlibDecoderBuffer:ByteArray;
private var _zlibDecoder:ZlibDecoder;
private var _socket:Socket;
private var _waitingResponseSize:uint = 0;

public function init( ip:String, port:int, policy:String, type:int, devConnection:Boolean = false
):void
{
_devConnection = devConnection;
_connectionSignal = new Signal(int);
_i = new BinaryInputStream();
_o = new BinaryInputStream();
_responseSignal = new Signal(IResponse);

Security.loadPolicyFile(policy);

_zlibDecoderBuffer = new ByteArray();
_zlibDecoderBuffer.endian = Endian.BIG_ENDIAN;
_zlibDecoder
```

```
= new ZlibDecoder();

_socket = new Socket();
_socket.endian = Endian.BIG_ENDIAN;

_socket.addEventListener(Event.CONNECT, onConnect);
_socket.addEventListener(ProgressEvent.SOCKET_DATA, onReceive);
_socket.addEventListener(Event.CLOSE, onClose);
_socket.addEventListener(IOErrorEvent.IO_ERROR, onError);
_socket.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);

_ip = ip;
_port = port;
_logger.info('Connecting to {0}:{1}', [_ip, _port]);

connectToServer();
}

public function connectToServer():void
{
//close the current connection and connect to the new server
if (_socket.connected)
{
_socket.readBytes(new ByteArray(), _socket.bytesAvailable);
_socket.close();
}

_zlibDecoder.reset(false, false);
_socket.connect(_ip, _port);
}

private function onConnect( e:Event ):void
{
_logger.info('Connected to {0}:{1}', [_ip, _port]);
_connectionSignal.dispatch(CONNECTED);
}

public function send( request:IRequest ):void
{
if (!active)
{
_logger.error("Cannot send message, not connected to server!");
return;
}

if(_simWrite>0)
{
_logger.error("This IO is already in use!");
return;
}

_simWrite++;
```

```
//clear out the old data
_i.clear();
request.write(_i);
_socket.writeShort(_i.length + 2);
_socket.writeBytes(_i);
_socket.flush();
ObjectPool.give(request);

_simWrite--;
}

private function onReceive( e:ProgressEvent ):void
{
if (_lockRead > 0)
return;

if(_simRead>0)
{
_logger.error("This IO is already in use!");
return;
}

_simRead++;

do
{
var size:uint;
if (_waitingResponseSize > 0)
{
size = _waitingResponseSize;
} else
{
_o.clear();
if (_socket.bytesAvailable >= 4)
{
size = _socket.readUnsignedInt() - 4;
_fullSize = size;
}
else
break;
}
if(_socket.bytesAvailable >= 62000)
{
_logger.error("This IO is broken!");
}
if (_socket.bytesAvailable >= 0)
{
var waitingByteSize:uint = Math.min(_socket.bytesAvailable,size);
_socket.readBytes(_o,
```

```
_o.bytesAvailable, waitingByteSize);
size -= waitingByteSize;

if(size<0)
{
_logger.error("TinCan - Critical input error!");
return;
}
if (size == 0)
{
//_socket.readBytes(_o, 0, size);
var protocolID:int = _o.readByte();
var speakerID:int = _o.readByte();
var header:int = _o.readByte();
var encoding:int = _o.readByte();
if (encoding == EncodingEnum.BINARYZLIBCOMPRESSED)
{
// Read compressed input into _zlibDecoderBuffer.
_zlibDecoderBuffer.length = _o.bytesAvailable + 4;
_zlibDecoderBuffer.position = 0;
_o.readBytes(_zlibDecoderBuffer,0, _o.bytesAvailable);

// Append the zlib sync-flush byte sequence.
_zlibDecoderBuffer.position = _zlibDecoderBuffer.length - 4;
_zlibDecoderBuffer.writeByte(0x00);
_zlibDecoderBuffer.writeByte(0x00);
_zlibDecoderBuffer.writeByte(0xff);
_zlibDecoderBuffer.writeByte(0xff);

// Uncompress the input to _io.
_zlibDecoderBuffer.position = 0;
_o.clear();
var bytesRead:uint = _zlibDecoder.feed(_zlibDecoderBuffer, _o);
if (bytesRead != _zlibDecoderBuffer.length || (_zlibDecoder.lastError !=
ZlibDecoderError.NEED_MORE_DATA && _zlibDecoder.lastError !=
ZlibDecoderError.NO_ERROR))
throw new Error("Failed to uncompress network message");

_o.position = 0;
encoding = EncodingEnum.BINARY;
}
if (_protocolListener == -1 || _protocolListener == protocolID ||
PacketFactory.isImportant(protocolID, header))
{
var ioLength:uint = _o.length;
var response:IResponse = PacketFactory.getResponse(_o, protocolID, header, encoding);
if (response)
{
_responseSignal.dispatch(response);
} else
{
```

```
_logger.error("Unknown Message Received {0}, {1}, {2}", [protocolID, speakerID, header]);
var msg:ProxyReportCrashRequest =
ProxyReportCrashRequest(_serverController.getRequest(ProtocolEnum.PROXY_CLIENT,
RequestEnum.PROXY_REPORT_CRASH));
msg.dataStr = 'Unknown Message Received ' + protocolID + ', ' + speakerID + ', ' + header;
_serverController.send(msg);
if (CONFIG::DEBUG)
throw new Error("Unknown Message Received");
}
}
_waitingResponseSize = 0;
} else
{
_waitingResponseSize = size;
//break;
}
}
} while (active && _socket.bytesAvailable > 0);

_simRead--;
}

private function unknownMessage( header:int ):void
{
_logger.error("Unknown Message Received {0}, {1}, {2}", [header, _socket.bytesAvailable,
_socket.readUTFBytes(_socket.bytesAvailable)]);
}

private function onClose( e:Event ):void
{
_logger.info('Connection to {0}:{1} closed', [_ip, _port]);
_connectionSignal.dispatch(CONNECTION_LOST);
}

protected function sendFailed( reason:String ):void
{
var myrequest:URLRequest = new URLRequest("https://" +
ExternalInterfaceAPI.getLoginHostname() + "/" + reason + CurrentUser.id);
myrequest.method = URLRequestMethod.POST;
var loader:URLLoader = new URLLoader();
loader.addEventListener(IOErrorEvent.IO_ERROR, postTo80Error, false, 0, true);
loader.load(myrequest);
_logger.info('sendFailed - Failed Message Sent To "https://{1}:80/{2}{3}',
[ExternalInterfaceAPI.getLoginHostname(), reason, CurrentUser.id]);
}

private function postTo80Error( e:IOErrorEvent ):void
{
e.stopImmediatePropagation();
}
```

```
private function onError( e:IOErrorEvent ):void
{
_logger.error('IOError on connection to {0}:{1} {2}', [_ip, _port, e.toString()]);

if (_devConnection && _port < 20020)
{
++_port;
connectToServer();
}

if (CONFIG::DEBUG)
_connectionSignal.dispatch(CONNECTED);
else
{
sendFailed("Failed-to-connect-to-proxy-ioerror-");
_connectionSignal.dispatch(CONNECTION_FAILED);
connectToServer();
}
}

private function onSecurityError( e:SecurityErrorEvent ):void
{
_logger.error('Security error on connection to {0}:{1} {2}', [_ip, _port, e.toString()]);
sendFailed("Failed-to-connect-to-proxy-securityerror-");
}

public function addConnectionListener( callback:Function ):void {
_connectionSignal.add(callback); }
public function removeConnectionListener( callback:Function ):void {
_connectionSignal.remove(callback); }

public function addResponseListener( callback:Function ):void { _responseSignal.add(callback);
}
public function removeResponseListener( callback:Function ):void {
_responseSignal.remove(callback); }

public function get active():Boolean { return (_socket) ? _socket.connected : false; }

public function set lockRead( v:Boolean ):void
{
_lockRead += v ? 1 : -1;
if (_lockRead == 0 && _socket && _socket.connected && _socket.bytesAvailable > 0)
onReceive(null);
}

public function get protocolListener():int { return _protocolListener; }
public function set protocolListener( v:int ):void { _protocolListener = v; }
public function set serverController( value:ServerController ):void { _serverController = value; }

public
```

```
function destroy():void
{
if (_socket.connected)
{
_socket.readBytes(new ByteArray(), _socket.bytesAvailable);
_socket.close();
}

_socket.removeEventListener(Event.CONNECT, onConnect);
_socket.removeEventListener(ProgressEvent.SOCKET_DATA, onReceive);
_socket.removeEventListener(Event.CLOSE, onClose);
_socket.removeEventListener(IOErrorEvent.IO_ERROR, onError);
_socket.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);

_zlibDecoder.dispose();
_zlibDecoder = null;
_zlibDecoderBuffer.clear();
_zlibDecoderBuffer = null;

_connectionSignal.removeAll();
_connectionSignal = null;
_o.clear();
_o = null;
_i.clear();
_i = null;
_responseSignal.removeAll();
_responseSignal = null;
_serverController = null;
_socket = null;
}

}
}

-----------------------------------
File 438: igw\com\service\server\connections\Connection.as
package com.service.server.connections
{
import com.Application;
import com.controller.ServerController;
import com.enum.ui.ButtonEnum;
import com.event.ServerEvent;
import com.model.asset.AssetModel;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.language.Localization;
import com.ui.alert.ConfirmationView;
import com.ui.core.ButtonPrototype;

import flash.events.HTTPStatusEvent;
import
```

```
flash.events.IEventDispatcher;
import flash.events.IOErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;

import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;

public class Connection
{
protected static var _instance:Connection;

protected static var MAX_CONNECTIONS:int = 50;

protected var _reconnectCount:int;
protected var _errorMessageShown:Boolean;
protected var _securityErrorFired:Boolean;
protected var _httpStatusCode:int;
protected var _assetModel:AssetModel;
protected var _eventDispatcher:IEventDispatcher;
protected var _serverController:ServerController;

public function Connection()
{
_instance = this;
}

public function connect():void
{
ExternalInterfaceAPI.logConsole("Imeprium Wrong Connection Method");
CurrentUser.id = Application.PLAYER_KEY;
CurrentUser.naid = Application.PLAYER_KABAM_NAID;
CurrentUser.authID = Application.PLAYER_TOKEN;
CurrentUser.oAuthID = Application.PLAYER_OAUTH;
CurrentUser.language = Application.LANGUAGE;
CurrentUser.country = Application.COUNTRY;
if (Application.NETWORK == Application.NETWORK_UNKNOWN)
Application.NETWORK = Application.NETWORK_KABAM;

connectToProxy();
}

protected function connectToProxy( isDev:Boolean = false ):void
{
ExternalInterfaceAPI.logConsole("Imperium connect to proxy");
Application.LANGUAGE = ExternalInterfaceAPI.getLocalizationTag();
if (Application.LANGUAGE != '' && Application.LANGUAGE != null)
Localization.instance.load('IMPG', Application.LANGUAGE);
else
```

```
Localization.instance.load('IMPG', 'en');
//ExternalInterfaceAPI.logConsole("Proxy: " + Application.PROXY_SERVER +":" +
Application.PROXY_PORT);
_serverController.connect(Application.PROXY_SERVER, Application.PROXY_PORT,
Application.PROXY_SERVER + ':843', isDev);
destroy();
}

protected function httpStatusHandler( e:HTTPStatusEvent ):void
{
_httpStatusCode = e.status;
}

protected function onIOError( e:IOErrorEvent ):void
{
sendFailed("Failed-to-Auth");
if (e.currentTarget.hasOwnProperty("data"))
{
var dataText:String = e.currentTarget.data;
if (dataText.indexOf("suspension") == -1 && dataText.indexOf("banned") == -1 &&
dataText.indexOf("imperium maintenance") == -1 && _httpStatusCode != 403 &&
_reconnectCount < MAX_CONNECTIONS)
{
++_reconnectCount;
_securityErrorFired = false;
connect();
}

if (!_errorMessageShown)
handleServerError(dataText);
}

e.stopImmediatePropagation();
}

protected function onSecurityError( e:IOErrorEvent ):void
{
_securityErrorFired = true;
sendFailed("Auth-Send-Failed-Security-Error");
}

protected function handleServerError( errorText:String ):void
{
_errorMessageShown = true;
var serverEvent:ServerEvent
errorText = errorText.toLowerCase();
if (errorText.indexOf('suspension') != -1)
serverEvent = new ServerEvent(ServerEvent.SUSPENSION);
else if (errorText.indexOf('banned') != -1)
serverEvent
```

```
= new ServerEvent(ServerEvent.BANNED);
else if (errorText.indexOf('imperium maintenance') != -1)
serverEvent = new ServerEvent(ServerEvent.MAINTENANCE);
else
serverEvent = new ServerEvent(ServerEvent.FAILED_TO_CONNECT);

_eventDispatcher.dispatchEvent(serverEvent);
}

protected function sendFailed( reason:String ):void
{
var loader:URLLoader = new URLLoader();
var myrequest:URLRequest = new URLRequest("https://" +
ExternalInterfaceAPI.getLoginHostname() + "/Reason=" + reason + '&http-Status-Code=' +
_httpStatusCode + '&Security-Error-Fired=' + _securityErrorFired);
var urlVariables:URLVariables = new URLVariables();
urlVariables.reason = reason;
myrequest.method = URLRequestMethod.GET;
myrequest.data = urlVariables;
loader = new URLLoader();
loader.addEventListener(IOErrorEvent.IO_ERROR, postTo80Error);
loader.load(myrequest);
}

private function postTo80Error( e:IOErrorEvent ):void
{
e.stopImmediatePropagation();
}

[Inject]
public function set assetModel( v:AssetModel ):void { _assetModel = v; }
[Inject]
public function set serverController( v:ServerController ):void { _serverController = v; }
[Inject]
public function set eventDispatcher( v:IEventDispatcher ):void { _eventDispatcher = v; }

public function destroy():void
{
_assetModel = null;
_instance = null;
_serverController = null;
}
}
}

-----------------------------------
File 439: igw\com\service\server\connections\DevConnection.as
package com.service.server.connections
{
import com.Application;
import
```

```
com.model.player.CurrentUser;

import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.ProgressEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class DevConnection extends Connection
{
private var _loader:URLLoader;

private static const _logger:ILogger = getLogger('DevConnection');

override public function connect():void
{
if (CONFIG::FLASH_LIVE_DEBUG_MODE == true)
{

//Security.loadPolicyFile("xmlsocket://37.18.201.89:843");

//var loginUri:String = "http://37.18.201.89:4000/login/xsolla";
/*CRYPTO/ Security.loadPolicyFile("xmlsocket://164.90.180.16:843");//*/
/*LIVE*/ Security.loadPolicyFile("xmlsocket://139.59.150.135:843");//*/

/*CRYPTO/ var loginUri:String = "http://164.90.180.16:4000/login/xsolla";//*/
/*LIVE*/ var loginUri:String = "http://139.59.150.135:4000/login/xsolla";//*/

_logger.info("login - Logging into {0}", [loginUri]);
//ExternalInterfaceAPI.logConsole("Imeprium Server URL = " + loginUri);
var variables:URLVariables = new URLVariables();
//variables["xsolla-token"] = copy token here!

variables["language"] = "en";
variables["country"] = "US";
variables["entry-tag"] = "";

//ExternalInterfaceAPI.logConsole("Imeprium Token = " +
ExternalInterfaceAPI.getPlayerXsollaGameAuthToken());
var request:URLRequest = new URLRequest(loginUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader
```

```
= new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);

_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(request);
}
else if (CONFIG::FLASH_DEBUG_MODE == true)
{
var defaultConnection:Object = _assetModel.getFromCache('data/DefaultConnection.txt');
Application.ASSET_PATH = "";//defaultConnection.assetPath;
/* CRYPTO */ //Application.PROXY_SERVER =
"164.90.180.16";//"134.209.251.89";//"165.227.132.33";//"37.18.201.91";//"127.0.0.1";//defaultConnection.s
/* TM */ Application.PROXY_SERVER =
"134.209.251.89";//"165.227.132.33";//"37.18.201.91";//"127.0.0.1";//defaultConnection.server;
/* STAGING */ //Application.PROXY_SERVER =
"165.227.132.33";//"165.227.132.33";//"37.18.201.91";//"127.0.0.1";//defaultConnection.server;
var variables:URLVariables = new URLVariables();
variables.fakeid = 1616;//7777;//16161616;//8001;//102;//37;//CurrentUser.naid;
variables.cacheavoidance = new Date().getTime(); // HACK: Prevent caching in IE.
///////var myrequest:URLRequest = new URLRequest('http://' + defaultConnection.server +
':4000/fakeloginflashdev');

/* CRYPTO */ //var myrequest:URLRequest = new
URLRequest('http://164.90.180.16:4000/fakeloginflashdev?fakeid='+variables.fakeid+'&proxyportoverride=
/* TM */ var myrequest:URLRequest = new
URLRequest('http://134.209.251.89:4000/fakeloginflashdev?fakeid='+variables.fakeid+'&proxyportoverride
/* STAGING */ //var myrequest:URLRequest = new
URLRequest('http://165.227.132.33:4000/fakeloginflashdev?fakeid='+variables.fakeid+'&proxyportoverride
//var myrequest:URLRequest = new
URLRequest('http://134.209.251.89:4000/fakeloginflashdev?fakeid='+variables.fakeid+'&proxyportoverride
//var myrequest:URLRequest = new
URLRequest('http://37.18.201.91:4000/fakeloginflashdev?fakeid='+variables.fakeid+'&proxyportoverride=2
//var myrequest:URLRequest = new
URLRequest('http://localhost:4000/fakeloginflashdev?fakeid='+variables.fakeid+'&proxyportoverride=2000
myrequest.method = URLRequestMethod.GET;
myrequest.data = variables;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(myrequest);

_logger.info("connect - Logging into http://{0}:4000/fakeloginflashdev", ["127.0.0.1"]);
}
else
{
var defaultConnection:Object = _assetModel.getFromCache('data/DefaultConnection.txt');
Application.ASSET_PATH
```

```
= defaultConnection.assetPath;
Application.PROXY_SERVER = defaultConnection.server;
var variables:URLVariables = new URLVariables();
variables.fakeid = CurrentUser.naid;
variables.cacheavoidance = new Date().getTime(); // HACK: Prevent caching in IE.
var myrequest:URLRequest = new URLRequest('http://' + defaultConnection.server +
':4000/fakeloginflashdev');
myrequest.method = URLRequestMethod.GET;
myrequest.data = variables;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(myrequest);

_logger.info("connect - Logging into http://{0}:4000/fakeloginflashdev",
[defaultConnection.server]);

}

}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");

var loginData:Object = JSON.parse(String(e.target.data));

var proxyIP:String = loginData["proxy-ip"];
if (proxyIP != "auto")
Application.PROXY_SERVER = proxyIP;

var proxyPortStr:String = loginData["proxy-port"];
var proxyPort:int = (proxyPortStr == "unknown") ? 20000 : int(proxyPortStr);
Application.PROXY_PORT = proxyPort;

CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginData["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];

connectToProxy(true);
}

override protected function onIOError( e:IOErrorEvent ):void
{
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
```

```
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

override public function destroy():void
{
super.destroy();
if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;
}
}
}

-----------------------------------
File 440: igw\com\service\server\connections\FacebookConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;

import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class FacebookConnection extends Connection
{
private var _loader:URLLoader;

private static const _logger:ILogger = getLogger('FacebookConnection');

override
```

```
public function connect():void
{
ExternalInterfaceAPI.logConsole("Imeprium Facebook Auth...");
Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");

var loginUri:String = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/facebook";
_logger.info("login - Logging into {0}", [loginUri]);
var variables:URLVariables = new URLVariables();
variables["facebook-user-id"] = ExternalInterfaceAPI.getFacebookUserId();
variables["facebook-access-token"] = ExternalInterfaceAPI.getFacebookUserAccessToken();
var request:URLRequest = new URLRequest(loginUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(request);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
var loginData:Object = JSON.parse(String(e.target.data));

Application.PROXY_SERVER = loginData["proxy-ip"];
Application.PROXY_PORT = loginData["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
Application.NETWORK = loginData["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginData["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];
Application.ASSET_PATH = loginData["asset-path"];

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);

connectToProxy();
} catch ( err:Error )
{
_logger.error("nAuthComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);
}
}
```

```
override protected function onIOError( e:IOErrorEvent ):void
{
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

override public function destroy():void
{
super.destroy();

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;
}
}
}

-------------------------------------
File 441: igw\com\service\server\connections\FacebookTokenConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;

import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public
```

```
class FacebookTokenConnection extends Connection
{
private var _loader:URLLoader;

private static const _logger:ILogger = getLogger('FacebookTokenConnection');

override public function connect():void
{
ExternalInterfaceAPI.logConsole("Imeprium Facebook Auth...");
Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");

var loginUri:String = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/token";
ExternalInterfaceAPI.logConsole(loginUri);
_logger.info("connect - Logging into {0}", [loginUri]);
var myrequest:URLRequest = new URLRequest(loginUri);
_logger.info("connect - Token = {0}", [Application.LOGIN_TOKEN]);
ExternalInterfaceAPI.logConsole(Application.LOGIN_TOKEN);
myrequest.method = URLRequestMethod.POST;
myrequest.data = Application.LOGIN_TOKEN;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(myrequest);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
var loginData:Object = JSON.parse(String(e.target.data));

Application.PROXY_SERVER = loginData["proxy-ip"];
Application.PROXY_PORT = loginData["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
Application.NETWORK = loginData["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginData["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];
Application.ASSET_PATH = loginData["asset-path"];

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);

connectToProxy();
}
```

```
catch ( err:Error )
{
_logger.error("nAuthComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);
}
}

override protected function onIOError( e:IOErrorEvent ):void
{
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

override public function destroy():void
{
super.destroy();

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;
}
}
}

------------------------------------
File 442: igw\com\service\server\connections\GuestConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.loading.LoadPriority;

import flash.display.Loader;
import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import
```

```
flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class GuestConnection extends Connection
{
private static const _logger:ILogger = getLogger('GuestConnection');

private var _loader:URLLoader;

public function GuestConnection()
{
}

override public function connect():void
{
Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");

var loginUri:String = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/guest";
_logger.info("login - Logging into {0}", [loginUri]);
var variables:URLVariables = new URLVariables();
variables["guest-user-id"] = ExternalInterfaceAPI.getGuestUserId();
variables["language"] = ExternalInterfaceAPI.getLanguageCode();
variables["country"] = ExternalInterfaceAPI.getCountryCode();
variables["entry-tag"] = ExternalInterfaceAPI.getEntryTag();
var request:URLRequest = new URLRequest(loginUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(request);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
var loginData:Object = JSON.parse(String(e.target.data));

ExternalInterfaceAPI.setGuestUserId(loginData["player-oauth-token"]);
```

```
Application.PROXY_SERVER = loginData["proxy-ip"];
Application.PROXY_PORT = loginData["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
Application.NETWORK = loginData["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = "";
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];
//Application.ASSET_PATH = loginData["asset-path"];

ExternalInterfaceAPI.setGuestUserId(loginData["player-oauth-token"]);

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);
ExternalInterfaceAPI.logConsole("Imperium Auth Successful!");

connectToProxy(true);

} catch ( err:Error )
{
_logger.error("nAuthComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);

ExternalInterfaceAPI.logConsole("Imperium auth failed cause of error!");
}
/*_logger.info('onAuthComplete')
try
{
var data:Object = JSON.parse(URLLoader(e.currentTarget).data);
var loginResponse:Object = data;

Application.PROXY_SERVER = loginResponse["proxy-ip"];
Application.PROXY_PORT = loginResponse["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginResponse["player-key"];
Application.NETWORK = loginResponse["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginResponse["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginResponse["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginResponse["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginResponse["language"];
CurrentUser.country = Application.COUNTRY = loginResponse["country"];
Application.ASSET_PATH = loginResponse["asset-path"];

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS,
```

```
httpStatusHandler);
}
_loader = null;

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);

connectToProxy();
} catch ( err:Error )
{
if (e.currentTarget.hasOwnProperty("data"))
{
handleServerError(e.currentTarget.data);
}
}*/
}

override protected function onIOError( e:IOErrorEvent ):void
{
//ExternalInterfaceAPI.logConsole("onIOError - Http status code");
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
//ExternalInterfaceAPI.logConsole("nSecurityError - Auth Send Failed!");
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

override public function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 443: igw\com\service\server\connections\KabamConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;

import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import
```

```
flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class KabamConnection extends Connection
{
private var _loader:URLLoader;

private static const _logger:ILogger = getLogger('KabamConnection');

override public function connect():void
{
Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");

var loginUri:String = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/token";
_logger.info("connect - Logging into {0}", [loginUri]);
var myrequest:URLRequest = new URLRequest(loginUri);
_logger.info("connect - Token = {0}", [Application.LOGIN_TOKEN]);
myrequest.method = URLRequestMethod.POST;
myrequest.data = Application.LOGIN_TOKEN;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(myrequest);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
var loginData:Object = JSON.parse(String(e.target.data));

Application.PROXY_SERVER = loginData["proxy-ip"];
Application.PROXY_PORT = loginData["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
Application.NETWORK = loginData["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginData["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];
Application.ASSET_PATH = loginData["asset-path"];

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);
```

```
connectToProxy();
} catch ( err:Error )
{
_logger.error("nAuthComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);
}
}

override protected function onIOError( e:IOErrorEvent ):void
{
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

override public function destroy():void
{
super.destroy();

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;
}
}
}

-------------------------------------
File 444: igw\com\service\server\connections\KongregateConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.kongregate.KongregateAPI;
import com.service.loading.LoadPriority;

import flash.display.Loader;
import
```

```
flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class KongregateConnection extends Connection
{
private static const _logger:ILogger = getLogger('KongregateConnection');

private var _kongregateAPI:KongregateAPI;
private var _loader:URLLoader;

override public function connect():void
{
ExternalInterfaceAPI.logConsole("Imperium Kongregate Auth...");
Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");

var loginUri:String = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/kongregate";
_logger.info("login - Logging into {0}", [loginUri]);
var variables:URLVariables = new URLVariables();
variables["kongregate-user-id"] = _kongregateAPI.userID;
variables["kongregate-user-auth-token"] = _kongregateAPI.gameAuthToken;
variables["language"] = ExternalInterfaceAPI.getLanguageCode();
variables["country"] = ExternalInterfaceAPI.getCountryCode();
variables["entry-tag"] = ExternalInterfaceAPI.getEntryTag();
var request:URLRequest = new URLRequest(loginUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(request);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info('onAuthComplete')
try
{
```

```
var data:Object = JSON.parse(URLLoader(e.currentTarget).data);
var loginResponse:Object = data;

Application.PROXY_SERVER = loginResponse["proxy-ip"];
Application.PROXY_PORT = loginResponse["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginResponse["player-key"];
Application.NETWORK = loginResponse["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginResponse["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginResponse["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginResponse["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginResponse["language"];
CurrentUser.country = Application.COUNTRY = loginResponse["country"];
Application.ASSET_PATH = loginResponse["asset-path"];

ExternalInterfaceAPI.logConsole("Imperium Auth Successful!");

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);

connectToProxy();
} catch ( err:Error )
{
ExternalInterfaceAPI.logConsole("Imperium Auth Failure!");

if (e.currentTarget.hasOwnProperty("data"))
{
handleServerError(e.currentTarget.data);
}
}
}

override protected function onIOError( e:IOErrorEvent ):void
{
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}
```

```
[Inject]
public function set kongregateAPI( v:KongregateAPI ):void { _kongregateAPI = v; }

override public function destroy():void
{
super.destroy();
_kongregateAPI = null;
}
}
}

-----------------------------------
File 445: igw\com\service\server\connections\SteamConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.loading.LoadPriority;

import flash.display.Loader;
import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class SteamConnection extends Connection
{
private static const _logger:ILogger = getLogger('SteamConnection');

private var _loader:URLLoader;

public function SteamConnection()
{
//ExternalInterfaceAPI.logConsole("Imperium Steam Connection Startup");
}

override public function connect():void
{
//ExternalInterfaceAPI.logConsole("Imperium Token Sending...");
Security.loadPolicyFile("xmlsocket://"
```

```
+ ExternalInterfaceAPI.getLoginHostname() + ":843");

var loginUri:String = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/steam";
_logger.info("login - Logging into {0}", [loginUri]);
//ExternalInterfaceAPI.logConsole("Imeprium Server URL = " + loginUri);
var variables:URLVariables = new URLVariables();
variables["steam-session-ticket"] = ExternalInterfaceAPI.getSteamSessionTicket();
variables["language"] = ExternalInterfaceAPI.getLanguageCode();
variables["country"] = ExternalInterfaceAPI.getCountryCode();
variables["entry-tag"] = ExternalInterfaceAPI.getEntryTag();
ExternalInterfaceAPI.logConsole("Imeprium Ticket = " +
ExternalInterfaceAPI.getSteamSessionTicket());
var request:URLRequest = new URLRequest(loginUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(request);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
var loginData:Object = JSON.parse(String(e.target.data));

Application.PROXY_SERVER = loginData["proxy-ip"];
Application.PROXY_PORT = loginData["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
Application.NETWORK = loginData["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginData["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];
Application.ASSET_PATH = loginData["asset-path"];

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);
//ExternalInterfaceAPI.logConsole("Imperium Auth Successful!");
connectToProxy();
} catch ( err:Error )
{
_logger.error("nAuthComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);
```

```
//ExternalInterfaceAPI.logConsole("Imperium auth failed cause of error!");
}
/*_logger.info('onAuthComplete')
try
{
var data:Object = JSON.parse(URLLoader(e.currentTarget).data);
var loginResponse:Object = data;

Application.PROXY_SERVER = loginResponse["proxy-ip"];
Application.PROXY_PORT = loginResponse["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginResponse["player-key"];
Application.NETWORK = loginResponse["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginResponse["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginResponse["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginResponse["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginResponse["language"];
CurrentUser.country = Application.COUNTRY = loginResponse["country"];
Application.ASSET_PATH = loginResponse["asset-path"];

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);

connectToProxy();
} catch ( err:Error )
{
if (e.currentTarget.hasOwnProperty("data"))
{
handleServerError(e.currentTarget.data);
}
}*/
}

override protected function onIOError( e:IOErrorEvent ):void
{
//ExternalInterfaceAPI.logConsole("onIOError - Http status code");
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
```

```
//ExternalInterfaceAPI.logConsole("nSecurityError - Auth Send Failed!");
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

override public function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 446: igw\com\service\server\connections\XsollaConnection.as
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.loading.LoadPriority;

import flash.display.Loader;
import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class XsollaConnection extends Connection
{
private static const _logger:ILogger = getLogger('XsollaConnection');

private var _loader:URLLoader;

private var _guestToXsolla:Boolean = false;

public function XsollaConnection()
{
ExternalInterfaceAPI.logConsole("Imperium Xsolla Connection Startup");
}

override public function connect():void
{
```

```
var defaultConnection:Object = _assetModel.getFromCache('data/DefaultConnection.txt');
if (!defaultConnection)
{
defaultConnection = new Object();
if ( CONFIG::IS_CRYPTO )
defaultConnection.server = '164.90.180.16'; // CRYPTO
else
defaultConnection.server = '139.59.150.135'; // LIVE
}
ExternalInterfaceAPI.logConsole("Imperium Token Sending...");
Security.loadPolicyFile("xmlsocket://" + defaultConnection.server + ":843");
//Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");


ExternalInterfaceAPI.logConsole("Connecting: " + defaultConnection.server);

var loginUri:String;
if(CONFIG::IS_DESKTOP){
loginUri = 'http://' + defaultConnection.server + ':4000/login/xsolla';
} else {
loginUri = ExternalInterfaceAPI.getLoginProtocol() + "://" +
ExternalInterfaceAPI.getLoginHostname() + ':' + ExternalInterfaceAPI.getLoginPort() +
"/login/xsolla";
}
_logger.info("login - Logging into {0}", [loginUri]);
//ExternalInterfaceAPI.logConsole("Imeprium Server URL = " + loginUri);
var variables:URLVariables = new URLVariables();
variables["xsolla-token"] = ExternalInterfaceAPI.getPlayerXsollaGameAuthToken();
variables["language"] = ExternalInterfaceAPI.getLanguageCode();
variables["country"] = ExternalInterfaceAPI.getCountryCode();
variables["entry-tag"] = ExternalInterfaceAPI.getEntryTag();

//TODO uncomment when all Guest Account is implemented
//var guestId:String = ExternalInterfaceAPI.getGuestToXsollaUserId();
//if(guestId.length > 0)
//{
// _guestToXsolla = true;
// variables["guest-user-id"] = guestId;
//}

//ExternalInterfaceAPI.logConsole("Imeprium Token = " +
ExternalInterfaceAPI.getPlayerXsollaGameAuthToken());
var request:URLRequest = new URLRequest(loginUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS,
```

```
httpStatusHandler);
_loader.load(request);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
var loginData:Object = JSON.parse(String(e.target.data));

Application.PROXY_SERVER = loginData["proxy-ip"];
Application.PROXY_PORT = loginData["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginData["player-key"];
Application.NETWORK = loginData["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginData["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginData["player-token"];
CurrentUser.oAuthID = Application.PLAYER_OAUTH = loginData["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginData["language"];
CurrentUser.country = Application.COUNTRY = loginData["country"];
//Application.ASSET_PATH = loginData["asset-path"];

if(_guestToXsolla)
{
ExternalInterfaceAPI.completeGuestToXsolla();
}

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);
//ExternalInterfaceAPI.logConsole("Imperium Auth Successful!");
connectToProxy();
} catch ( err:Error )
{
_logger.error("nAuthComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);

//ExternalInterfaceAPI.logConsole("Imperium auth failed cause of error!");
}
/*_logger.info('onAuthComplete')
try
{
var data:Object = JSON.parse(URLLoader(e.currentTarget).data);
var loginResponse:Object = data;

Application.PROXY_SERVER = loginResponse["proxy-ip"];
Application.PROXY_PORT = loginResponse["proxy-port"];
CurrentUser.id = Application.PLAYER_KEY = loginResponse["player-key"];
Application.NETWORK = loginResponse["play-platform"];
CurrentUser.naid = Application.PLAYER_KABAM_NAID = loginResponse["kabam-naid"];
CurrentUser.authID = Application.PLAYER_TOKEN = loginResponse["player-token"];
CurrentUser.oAuthID
```

```
= Application.PLAYER_OAUTH = loginResponse["player-oauth-token"];
CurrentUser.language = Application.LANGUAGE = loginResponse["language"];
CurrentUser.country = Application.COUNTRY = loginResponse["country"];
Application.ASSET_PATH = loginResponse["asset-path"];

if (_loader)
{
_loader.removeEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.removeEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.removeEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.removeEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
}
_loader = null;

ExternalInterfaceAPI.setPlayPlatform(Application.NETWORK);

connectToProxy();
} catch ( err:Error )
{
if (e.currentTarget.hasOwnProperty("data"))
{
handleServerError(e.currentTarget.data);
}
}*/
}

override protected function onIOError( e:IOErrorEvent ):void
{
//ExternalInterfaceAPI.logConsole("onIOError - Http status code");
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
//ExternalInterfaceAPI.logConsole("nSecurityError - Auth Send Failed!");
_logger.error('onSecurityError - Auth Send Failed');
super.onSecurityError(e);
}

//[Inject]
//public function set xsollaAPI( v:XsollaAPI ):void { _xsollaAPI = v; }

override public function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
```

File 447: igw\com\service\server\connections\XsollaPaymentConnection.as

```
package com.service.server.connections
{
import com.Application;
import com.model.player.CurrentUser;
import com.service.ExternalInterfaceAPI;
import com.service.loading.LoadPriority;

import flash.display.Loader;
import flash.events.Event;
import flash.events.HTTPStatusEvent;
import flash.events.IOErrorEvent;
import flash.events.SecurityErrorEvent;
import flash.net.URLLoader;
import flash.net.URLRequest;
import flash.net.URLRequestMethod;
import flash.net.URLVariables;
import flash.system.Security;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

public class XsollaPaymentConnection extends Connection
{

public function XsollaPaymentConnection()
{
//ExternalInterfaceAPI.logConsole("Imperium Xsolla Paymment Connection Startup");
}
private static const _logger:ILogger = getLogger('XsollaPayment');

private var _loader:URLLoader;

override public function connect():void
{

var defaultConnection:Object = _assetModel.getFromCache('data/DefaultConnection.txt');

if (!defaultConnection)
{
defaultConnection = new Object();
if ( CONFIG::IS_CRYPTO )
defaultConnection.server = '164.90.180.16'; // CRYPTO
else
defaultConnection.server = '139.59.150.135'; // LIVE
}
//ExternalInterfaceAPI.logConsole("Imperium Payment Token Request...");
Security.loadPolicyFile("xmlsocket://" + defaultConnection.server + ":843");
//Security.loadPolicyFile("xmlsocket://" + ExternalInterfaceAPI.getLoginHostname() + ":843");

var
```

```
paymentUri:String = ExternalInterfaceAPI.getPaymentProtocol() + "://" +
ExternalInterfaceAPI.getPaymentHostname() + ':' + ExternalInterfaceAPI.getPaymentPort() +
"/token/xsolla";
if(CONFIG::IS_DESKTOP){
paymentUri = 'http://' + defaultConnection.server + ':9000/token/xsolla';
} else {
paymentUri = ExternalInterfaceAPI.getPaymentProtocol() + "://" +
ExternalInterfaceAPI.getPaymentHostname() + ':' + ExternalInterfaceAPI.getPaymentPort() +
"/token/xsolla";
}

_logger.info("payment - Logging into {0}", [paymentUri]);
var variables:URLVariables = new URLVariables();
variables["xsolla-token"] = ExternalInterfaceAPI.getPlayerXsollaGameAuthToken();

var request:URLRequest = new URLRequest(paymentUri);
request.data = variables;
request.method = URLRequestMethod.POST;
_loader = new URLLoader();
_loader.addEventListener(Event.COMPLETE, onPlayerDataReceived);
_loader.addEventListener(IOErrorEvent.IO_ERROR, onIOError);
_loader.addEventListener(SecurityErrorEvent.SECURITY_ERROR, onSecurityError);
_loader.addEventListener(HTTPStatusEvent.HTTP_STATUS, httpStatusHandler);
_loader.load(request);
}

private function onPlayerDataReceived( e:Event ):void
{
_logger.info("onPlayerDataReceived");
try
{
//ExternalInterfaceAPI.logConsole("Imperium Payment Successful!");

//var paymentData:Object = JSON.parse(String(e.target.data));
//var access_token:String = paymentData["token"];
var access_token:String = String(e.target.data)

ExternalInterfaceAPI.openXsollaStore(access_token);

} catch ( err:Error )
{
_logger.error("nPaymentComplete - Failed");
if (!_errorMessageShown && e.currentTarget.hasOwnProperty("data"))
handleServerError(e.currentTarget.data);

ExternalInterfaceAPI.logConsole("Imperium Payment failed cause of error!");
}
}

override protected function onIOError( e:IOErrorEvent ):void
{
```

```
//ExternalInterfaceAPI.logConsole("onIOError - Http status code");
_logger.error('onIOError - Http status code - {0}', [_httpStatusCode]);
super.onIOError(e);
destroy();
}

override protected function onSecurityError( e:IOErrorEvent ):void
{
//ExternalInterfaceAPI.logConsole("nSecurityError - Payment Send Failed!");
_logger.error('onSecurityError - Payment Send Failed');
super.onSecurityError(e);
destroy();
}
}
}
```

File 448: igw\com\service\server\incoming\TransactionResponse.as

```
package com.service.server.incoming
{
import com.model.transaction.TransactionVO;
import com.service.server.BinaryInputStream;
import com.service.server.ITransactionResponse;

import org.shared.ObjectPool;

public class TransactionResponse implements ITransactionResponse
{
protected var _data:TransactionVO;

protected var _header:int;
protected var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
_data = ObjectPool.get(TransactionVO);
_data.read(input);
}

public function readJSON( data:Object ):void
{
_data = ObjectPool.get(TransactionVO);
_data.readJSON(data);
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public
```

```
function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get success():Boolean { return _data.success; }
public function get token():int { return _data.token; }

public function get data():TransactionVO { return _data; }
public function set data( v:TransactionVO ):void { _data = v; }

public function destroy():void
{
_data = null;
}
}
}

-----------------------------------
File 449: igw\com\service\server\incoming\alliance\AllianceBaselineResponse.as
package com.service.server.incoming.alliance
{
import com.model.alliance.AllianceVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class AllianceBaselineResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var alliance:AllianceVO;

public var currentUserAllianceKey:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
input.checkToken();
var key:String = input.readUTF(); // key
input.checkToken();

var name:String = input.readUTF(); // name
var factionPrototype:IPrototype =
PrototypeModel.instance.getFactionPrototypeByName(input.readUTF()); // factionPrototype
var motd:String = input.readUTF(); // motd
var description:String = input.readUTF(); // description
var isPublic:Boolean = input.readBoolean(); // publicAlliance


input.checkToken();
```

```
// data about the player himself
CurrentUser.alliance = input.readUTF(); // your alliance key
CurrentUser.allianceRank = input.readInt(); // your rank
CurrentUser.isAllianceOpen = input.readBoolean(); // your alliance is public
CurrentUser.allowAllianceInvites = input.readBoolean(); // ignore invites

input.checkToken();

alliance = new AllianceVO(key, name, factionPrototype, motd, description, isPublic);
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
alliance = null;
}
}
}

-----------------------------------
File 450: igw\com\service\server\incoming\alliance\AllianceGenericResponse.as
package com.service.server.incoming.alliance
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class AllianceGenericResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var responseEnum:int;
public var allianceKey:String;

public function read( input:BinaryInputStream ):void
{
```

```
input.checkToken();
responseEnum = input.readInt(); // response code enum
allianceKey = input.readUTF(); // alliance key
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}
```

File 451: igw\com\service\server\incoming\alliance\AllianceInviteResponse.as

```
package com.service.server.incoming.alliance
{
import com.model.alliance.AllianceInviteVO;
import com.model.alliance.AllianceVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class AllianceInviteResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var inviteVO:AllianceInviteVO;

public function read( input:BinaryInputStream ):void
{
var factionPrototype:IPrototype =
PrototypeModel.instance.getFactionPrototypeByName(CurrentUser.faction);
input.checkToken();
```

```
var key:String = input.readUTF(); // alliance key
var allianceName:String = input.readUTF(); // alliance name
var memberCount:int = input.readInt(); // num members
var inviterKey:String = input.readUTF(); // inviter key
var inviterName:String = input.readUTF(); // inviter name
input.checkToken();
var alliance:AllianceVO = new AllianceVO(key, allianceName, factionPrototype, '', '', false);
alliance.memberCount = memberCount;
inviteVO = new AllianceInviteVO(inviterKey, inviterName, alliance);
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
inviteVO = null;
}
}
}

-------------------------------------
File 452: igw\com\service\server\incoming\alliance\AllianceRosterResponse.as
package com.service.server.incoming.alliance
{
import com.model.alliance.AllianceMemberVO;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class AllianceRosterResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var allianceKey:String;
public var members:Vector.<AllianceMemberVO>;

public function read( input:BinaryInputStream ):void
{
```

```
members = new Vector.<AllianceMemberVO>;
input.checkToken();
allianceKey = input.readUTF(); // alliance key
var member:AllianceMemberVO;
var numRoster:int = input.readUnsignedInt();
for (var i:int = 0; i < numRoster; ++i)
{
input.checkToken();
member = new AllianceMemberVO(input.readUTF(), input.readUTF(), input.readInt64(),
input.readInt64(), input.readInt());
input.checkToken();
members.push(member);
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 453: igw\com\service\server\incoming\alliance\PublicAlliancesResponse.as
package com.service.server.incoming.alliance
{
import com.model.alliance.AllianceVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class PublicAlliancesResponse implements IResponse
{
private
```

```
var _header:int;
private var _protocolID:int;

public var alliances:Vector.<AllianceVO>;

public function read( input:BinaryInputStream ):void
{
alliances = new Vector.<AllianceVO>;
input.checkToken();
var alliance:AllianceVO;
var num:int = input.readUnsignedShort();
var factionPrototype:IPrototype =
PrototypeModel.instance.getFactionPrototypeByName(CurrentUser.faction);
for (var i:int = 0; i < num; ++i)
{
input.checkToken();
var key:String = input.readUTF(); // key
var name:String = input.readUTF(); // name
var membercount:int = input.readInt();
var motd:String = input.readUTF(); // motd
var description:String = input.readUTF(); // description

input.checkToken();

alliance = new AllianceVO(key, name, factionPrototype, motd, description, true);
alliance.memberCount = membercount;
alliances.push(alliance);
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}
```

File 454: igw\com\service\server\incoming\battle\BattleDataResponse.as

```
package com.service.server.incoming.battle
{
import com.enum.server.ResponseEnum;
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;

public class BattleDataResponse implements ITickedResponse
{
// ITickedReponse stuff
private var _header:int;
private var _protocolID:int;
public var _tick:int;
public var _timeStep:int;

private var _data:BinaryInputStream = new BinaryInputStream;

public function read( input:BinaryInputStream ):void
{
input.readBytes( _data, 0, input.bytesAvailable - 8 );
_tick = input.readInt();
_timeStep = input.readInt();
_data.sequenceToken = input.sequenceToken;
_data.setStringInputCache( input.battleStringInputCache );
// the final checkToken can't be done yet since there may be other tokens embedded in the
binary data
}

public function readJSON( data:Object ):void
{
throw new Error("this is unsupported");
}

public function get input():BinaryInputStream { return _data; }

public function get isBaseline():Boolean { return _header ==
ResponseEnum.BATTLE_BASELINE; }
public function get isTicked():Boolean { return true; }
public function get addTick():Boolean { return true; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public
```

```
function destroy():void {}

}
}


------------------------------------
File 455: igw\com\service\server\incoming\battle\BattleDebugLinesResponse.as
package com.service.server.incoming.battle
{
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;
import com.service.server.incoming.data.DebugLineData;
import com.service.server.incoming.data.IServerData;
import com.service.server.incoming.data.RemovedObjectData;

import org.shared.ObjectPool;

public class BattleDebugLinesResponse implements ITickedResponse
{
public var debugLines:Vector.<DebugLineData> = new Vector.<DebugLineData>;
public var removedLines:Vector.<RemovedObjectData> = new Vector.<RemovedObjectData>;

private var _header:int;
private var _protocolID:int;
private var _tick:int;
private var _timeStep:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
_tick = input.readInt();
_timeStep = input.readInt();

var debugLine:DebugLineData;
var numAdded:int = input.readUnsignedInt();
for (var i:int = 0; i < numAdded; i++)
{
input.readUnsignedInt();
debugLine = ObjectPool.get(DebugLineData);
debugLine.read(input);
debugLines.push(debugLine);
}
var data:IServerData;
var numRemoved:int = input.readUnsignedInt();
for (i = 0; i < numRemoved; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).id = "DebugLine" + input.readUnsignedInt();
RemovedObjectData(data).reason = input.readInt();
removedLines.push(data);
```

```
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isBaseline():Boolean { return false; }
public function get isTicked():Boolean { return true; }
public function get addTick():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public function destroy():void
{
var i:int = 0;
for (i = 0; i < debugLines.length; i++)
ObjectPool.give(debugLines[i]);
debugLines.length = 0;

for (i = 0; i < removedLines.length; i++)
ObjectPool.give(removedLines[i]);
removedLines.length = 0;
}
}
}

-------------------------------------
File 456: igw\com\service\server\incoming\battle\BattleHasEndedResponse.as
package com.service.server.incoming.battle
{
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;

import flash.utils.Dictionary;
import com.service.server.incoming.data.CrewMemberData;
import org.shared.ObjectPool;

public class BattleHasEndedResponse implements ITickedResponse
{
public var battleKey:String;
public
```

```
var victors:Dictionary;
public var alloyLoot:Number;
public var energyLoot:Number;
public var syntheticLoot:Number;
public var creditBounty:Number;
public var blueprintReward:String;
public var crewReward:CrewMemberData;
public var cargoFull:Boolean;

private var _header:int;
private var _protocolID:int;
private var _tick:int;
private var _timeStep:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
_tick = input.readInt();
battleKey = input.readUTF();
victors = new Dictionary();
var numVictors:int = input.readUnsignedInt();
for (var i:int = 0; i < numVictors; i++)
{
var key:String = input.readUTF();
victors[key] = true;
}
alloyLoot = input.readInt64();
energyLoot = input.readInt64();
syntheticLoot = input.readInt64();
creditBounty = input.readInt64();
blueprintReward = input.readUTF();
var hasCrew:Boolean = false;
hasCrew = input.readBoolean();
if (hasCrew)
{
crewReward = ObjectPool.get(CrewMemberData);
crewReward.read(input);
}
cargoFull = input.readBoolean();
input.checkToken();
_tick = int.MAX_VALUE; // process this on the last tick
}

public function readJSON( data:Object ):void
{
_tick = data.battleEndTick;
_timeStep = data.tickTimeMillis;

victors = data.victors;
}

public
```

```
function get isBaseline():Boolean { return false; }
public function get isTicked():Boolean { return true; }
public function get addTick():Boolean { return true; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public function destroy():void
{
}
}
}

-------------------------------------
File 457: igw\com\service\server\incoming\battle\BattleParticipantInfo.as
package com.service.server.incoming.battle
{
import com.service.server.BinaryInputStream;

public class BattleParticipantInfo
{
public var id:String;
public var level:int;

public function read( input:BinaryInputStream ):void
{
id = input.readUTF();
level = input.readInt();
}
}
}
-------------------------------------
File 458: igw\com\service\server\incoming\battlelog\BattleLogDetailsResponse.as
package com.service.server.incoming.battlelog
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.BattleLogPlayerDetailInfo;

public class BattleLogDetailsResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public
```

```
var battleKey:String;
public var winners:Vector.<BattleLogPlayerDetailInfo>;
public var losers:Vector.<BattleLogPlayerDetailInfo>;
public var startTime:Number;
public var endTime:Number;

public function BattleLogDetailsResponse()
{
winners = new Vector.<BattleLogPlayerDetailInfo>;
losers = new Vector.<BattleLogPlayerDetailInfo>;
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();
var i:uint;
battleKey = input.readUTF();

var currentPlayer:BattleLogPlayerDetailInfo;
var winnersLength:uint = input.readUnsignedInt();
for(; i < winnersLength; ++i)
{
currentPlayer = new BattleLogPlayerDetailInfo();
currentPlayer.read(input);
winners.push(currentPlayer);
}

var losersLength:uint = input.readUnsignedInt();
for(i = 0; i < losersLength; ++i)
{
currentPlayer = new BattleLogPlayerDetailInfo();
currentPlayer.read(input);
losers.push(currentPlayer);
}

startTime = input.readInt64();
endTime = input.readInt64();

input.checkToken();
}

public function readJSON(data:Object):void
{
battleKey = data._id;
startTime = data.startTime;
endTime = data.endTime;
for each( var playerJson:Object in data.playerResults )
{
var playerSummary:BattleLogPlayerDetailInfo = new BattleLogPlayerDetailInfo();
playerSummary.readJSON( playerJson );
if(
```

```
playerJson.victor == true )
{
winners.push(playerSummary);
}
else
{
losers.push(playerSummary);
}
}

}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 459: igw\com\service\server\incoming\battlelog\BattleLogListResponse.as

```
package com.service.server.incoming.battlelog
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.BattleLogPlayerSummaryInfo;
import com.service.server.incoming.data.BattleLogSummaryInfo;

public class BattleLogListResponse implements IResponse
{
public var battles:Vector.<BattleLogSummaryInfo>;

private var _header:int;
private var _protocolID:int;

public function BattleLogListResponse()
{
battles = new Vector.<BattleLogSummaryInfo>;
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();
var currentBattle:BattleLogSummaryInfo;
var
```

```
battleLogsLength:uint = input.readUnsignedShort();
for(var i:uint = 0; i < battleLogsLength; ++i)
{
currentBattle = new BattleLogSummaryInfo();
currentBattle.read(input);
battles.push(currentBattle);
}
input.checkToken();
}

public function readJSON(input:Object):void
{
for each (var summaryJson:Object in input)
{
var currentBattle:BattleLogSummaryInfo = new BattleLogSummaryInfo();
currentBattle.readJSON( summaryJson );
battles.push(currentBattle);
}
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}



}
}
```

File 460: igw\com\service\server\incoming\chat\ChatBaselineResponse.as

```
package com.service.server.incoming.chat
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import com.model.player.CurrentUser;

public class ChatBaselineResponse implements IResponse
{
public var ignoredPlayers:Vector.<String> = new Vector.<String>;
private var _header:int;
private var _protocolID:int;

public
```

```
function read( input:BinaryInputStream ):void
{
var i:int = 0;
var numIgnores:int = input.readUnsignedInt();
for(i=0; i<numIgnores; i++)
{
ignoredPlayers.push(input.readUTF());
}

CurrentUser.allianceRank = input.readInt(); // your rank
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in ChatBaselineResponse is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
ignoredPlayers.length = 0;
}
}
}
```

File 461: igw\com\service\server\incoming\chat\ChatEventResponse.as

```
package com.service.server.incoming.chat
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class ChatEventResponse implements IResponse
{
public var attacker:String;
public var target:String;
public var sector:String;
public var locationX:Number;
public var locationY:Number;

private var _header:int;
private var _protocolID:int;

public
```

```
function read( input:BinaryInputStream ):void
{
input.checkToken();
var type:int = input.readInt();
if(type == 1)
{
attacker = input.readUTF();
target = input.readUTF();
sector = input.readUTF();
locationX = input.readDouble();
locationY = input.readDouble();
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 462: igw\com\service\server\incoming\chat\ChatResponse.as

```
package com.service.server.incoming.chat
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class ChatResponse implements IResponse
{
public var responseCode:int;
public var channel:int;
public var senderKey:String;
public var senderName:String;
public var senderFaction:String;
public var message:String;

private var _header:int;
private var _protocolID:int;

public
```

```
function read( input:BinaryInputStream ):void
{
input.checkToken();
responseCode = input.readInt();
channel = input.readInt();
senderKey = input.readUTF();
senderName = input.readUTF();
senderFaction = input.readUTF();
message = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 463: igw\com\service\server\incoming\data\AchievementData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class AchievementData implements IServerData
{
public var key:String;
public var achievementPrototype:String;
public var claimedFlag:Boolean;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
key = input.readUTF();
input.checkToken();
achievementPrototype = input.readUTF();
claimedFlag = input.readBoolean();
input.readBoolean();
```

```
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in AchievementData is not supported");
}

public function destroy():void
{
}
}
}

-----------------------------------
File 464: igw\com\service\server\incoming\data\ActiveDefenseData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class ActiveDefenseData extends ModuleData
{

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
super.read(input);
input.checkToken();
}

override public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

override public function destroy():void
{

}
}
}
```

File 465: igw\com\service\server\incoming\data\ActiveDefenseHitData.as
```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class ActiveDefenseHitData implements IServerData
{
public var attachPoint:String;
public var owningShip:String;
public var target:String;

public function read( input:BinaryInputStream ):void
{
//Target here is guaranteed to be unique, and is intended to be used when an attack is removed with reason "intercepted" to determine who shot it down.
//In C++-land, this is a map of target : attachPoint/owningShip objects.
target = "Attack" + input.readUnsignedInt();
attachPoint = input.readUTF();
owningShip = input.readUTF();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in ActiveDefenseHitData is not supported");
}

public function destroy():void
{

}
}
}
```

File 466: igw\com\service\server\incoming\data\AreaAttackData.as
```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class AreaAttackData extends AttackData
{
public var endX:Number;
public var endY:Number;
public var reloadTime:Number;
public var finishTick:int;
public var nextFireTick:int;
public var sourceAttachPoint:String;

override
```

```
public function read( input:BinaryInputStream ):void
{
input.checkToken();
super.read(input);
endX = input.readDouble();
endY = input.readDouble();
reloadTime = input.readDouble();
finishTick = input.readInt();
nextFireTick = input.readInt();
sourceAttachPoint = input.readUTF();
input.checkToken();
}

override public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

override public function destroy():void
{

}
}
}

-----------------------------------
File 467: igw\com\service\server\incoming\data\AreaAttackHitData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class AreaAttackHitData implements IServerData
{
public var attackId:String;
public var target:String
public var locationX:Number;
public var locationY:Number;

public function read(input:BinaryInputStream):void
{
attackId = String(input.readUnsignedInt());
target
```

```
= input.readUTF();

locationX = input.readDouble();
locationY = input.readDouble();
}

public function readJSON(data:Object):void
{
}

public function destroy():void
{
}
}
}
```

File 468: igw\com\service\server\incoming\data\AttackData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import com.service.server.replicable.ReplicableStruct;

import flash.geom.Point;

public class AttackData extends ReplicableStruct implements IServerData
{
public var playerOwnerId:String;
public var entityOwnerId:String;
public var id:int; // raw int Id used by replication
private var _attackId:String; // "Attack" prefixed Id used externally
public var start:Point;
public var startTick:int;
public var subsystemTarget:int;
public var targetPointIndex:int;
public var targetEntityId:String;
public var weaponPrototype:String;
public var location:Point;
public var _rotation:Number;

public override function read( input:BinaryInputStream ):void
{
input.checkToken();
playerOwnerId = input.readUTF();
entityOwnerId = input.readUTF();
id = input.readUnsignedInt();
attackId = String(id);
start = readLocation(input);
startTick = input.readInt();
targetEntityId = input.readUTF();
weaponPrototype = input.readUTF();
subsystemTarget
```

```
= input.readInt();
rotation = input.readUnsignedByte();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function destroy():void
{

}

public function set rotation( value256:Number ):void
{
_rotation = value256 * Math.PI / 128;
}

public function get rotation():Number
{
return _rotation;
}

public function set attackId( value:String ):void
{
_attackId = "Attack" + value;
}

public function get attackId():String
{
return _attackId;
}


}
}

-------------------------------------
File 469: igw\com\service\server\incoming\data\BaseData.as
package com.service.server.incoming.data
{
import com.model.player.CurrentUser;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

import org.shared.ObjectPool;

public class BaseData implements IServerData
{
public
```

```
var alloy:uint;
public var credits:uint;
public var energy:uint;
public var id:String;
public var lastUpdateTimeUTCMillis:Number;
public var bubbleEnds:Number;
public var bubbleTimeRemaining:Number;
public var ownerID:String;
public var sector:SectorData;
public var sectorLocationX:int;
public var sectorLocationY:Number;
public var synthetic:uint;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

input.checkToken();
id = input.readUTF();
input.checkToken();

sector = ObjectPool.get(SectorData);
sector.id = input.readUTF();

ownerID = input.readUTF();

alloy = input.readInt64();
synthetic = input.readInt64();
energy = input.readInt64();
credits = input.readInt64();

lastUpdateTimeUTCMillis = input.readInt64();
bubbleEnds = input.readInt64();

CurrentUser.baseRating = input.readInt();
input.readInt(); // baseRatingTech - currently we don't use it in Client

sectorLocationX = input.readDouble();
sectorLocationY = input.readDouble();
sector.prototype = PrototypeModel.instance.getSectorPrototypeByName(input.readUTF());
sector.appearanceSeed = input.readInt();

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in BaseData is not supported");
}

public
```

```
function destroy():void
{
}
}
}

------------------------------------
File 470: igw\com\service\server\incoming\data\BattleData.as
package com.service.server.incoming.data
{
import com.model.player.PlayerVO;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.battle.BattleDataResponse;
import com.service.server.incoming.battle.BattleParticipantInfo;
import com.service.server.replicable.ReplicableSet;
import com.service.server.replicable.ReplicableStruct;

import flash.utils.Dictionary;

import org.shared.ObjectPool;

public class BattleData extends ReplicableStruct implements IResponse
{
public static var globalInstance:BattleData = new BattleData;

public var tick:int;
public var timeStep:int;

public var hasBeenBaselined:Boolean = false;

// serialized data
public var activeSplitPrototypes:Vector.<String> = new Vector.<String>;
public var battleKey:String;
public var sector:SectorData;

public var maxSizeX:int;
public var maxSizeY:int;

public var galacticName:String;
public var backgroundId:int;
public var planetId:int;
public var moonQuantity:int;
public var asteroidQuantity:int;
public var appearanceSeed:int;


public var battleStartTick:int;
public var battleEndTick:int;
public
```

```
var loadTimeoutTick:int;
public var _battleState:int;
public var battleStateChanged:Boolean;
public var isBaseCombat:Boolean;
public var isInstancedMission:Boolean;
public var baseOwner:String;
public var alloy:Number;
public var synthetic:Number;
public var energy:Number;
public var credits:Number;
public var missionPersistence:String;
public var players:ReplicableSet = new ReplicableSet;
public var entities:ReplicableSet = new ReplicableSet;
public var deadEntities:ReplicableSet = new ReplicableSet;
public var areaAttacks:ReplicableSet = new ReplicableSet;
public var beamAttacks:ReplicableSet = new ReplicableSet;
public var droneAttacks:ReplicableSet = new ReplicableSet;
public var projectileAttacks:ReplicableSet = new ReplicableSet;
public var participants:ReplicableSet = new ReplicableSet;

public var areaAttackHits:Vector.<AreaAttackHitData> = new Vector.<AreaAttackHitData>;
public var adMisses:Array; // unused for now
public var adHits:Object = new Object();

public function readAreaAttackHits( input:BinaryInputStream ):Vector.<AreaAttackHitData>
{
areaAttackHits.length = 0;
var numAreaCollisions:int = input.readUnsignedInt();
for (var i:int = 0; i < numAreaCollisions; i++)
{
input.checkToken();
//Data on area attacks that have made contact, do something with this too.
var data:AreaAttackHitData = ObjectPool.get(AreaAttackHitData);
data.attackId = "Attack" + input.readUnsignedInt();
data.target = input.readUTF();
data.locationX = input.readDouble();
data.locationY = input.readDouble();
areaAttackHits.push(data);
input.checkToken();
}
return areaAttackHits;
}

public function readADMisses( input:BinaryInputStream ):void
{
var numADMisses:int = input.readUnsignedInt();
for (var i:int = 0; i < numADMisses; i++)
{
//Target here may not be unique, multiple attempts may have been made on the same target.
//In C++-land this is a vector of target/attachPoint/owningShip objects
var
```

```
target:uint = input.readUnsignedInt();
var attachPoint:String = input.readUTF();
var owningShip:String = input.readUTF();
}
}

public function readADHits( input:BinaryInputStream ):Object
{
var adHit:ActiveDefenseHitData;
var numADHits:int = input.readUnsignedInt();
for (var i:int = 0; i < numADHits; i++)
{
adHit = ObjectPool.get(ActiveDefenseHitData);
adHit.read(input);
if (adHit.attachPoint != "")
adHits[adHit.target] = adHit;
else
ObjectPool.give(adHit);
}
return adHits;
}

public function decodeResponse( response:BattleDataResponse ):int
{
// clear out old deltas
battleStateChanged = false;
players.resetDeltas();
for (var i:int = 0; i < entities.modified.length; ++i)
{
entities.modified[i].resetDeltas();
}
entities.resetDeltas();

deadEntities.resetDeltas();
areaAttacks.resetDeltas();
beamAttacks.resetDeltas();
droneAttacks.resetDeltas();
projectileAttacks.resetDeltas();
participants.resetDeltas();
areaAttackHits.length = 0;
adMisses = null;

// start reading stuff
tick = response.tick;
timeStep = response.timeStep;
if (response.isBaseline)
{
response.input.readStringCacheBaseline();
}
decode(response.input);
response.input.readUnsignedInt();
```

```
// this is the data modification number, which the flash client doesn't (yet) track
return 0;
}


private function readPlayerKey( input:BinaryInputStream ):String
{
return input.readUTF();
}

private function readAttackKey( input:BinaryInputStream ):int
{
return input.readInt();
}

private function readRemovedObjectData( input:BinaryInputStream ):RemovedObjectData
{
var removedData:RemovedObjectData = ObjectPool.get(RemovedObjectData);
removedData.read(input);
return removedData;
}

private function readRemovedAttackData( input:BinaryInputStream ):RemovedAttackData
{
var removedData:RemovedAttackData = ObjectPool.get(RemovedAttackData);
removedData.read(input);
return removedData;
}

public function BattleData()
{
players.readKey = readPlayerKey;
players.readRemove = readRemovedObjectData;
players.elementType = PlayerVO;

entities.readKey = readPlayerKey;
entities.readRemove = readRemovedObjectData;
entities.elementType = BattleEntityData;

deadEntities.readKey = readPlayerKey;
deadEntities.readRemove = readRemovedObjectData;
deadEntities.elementType = BattleEntityData;

areaAttacks.readKey = readAttackKey;
areaAttacks.readRemove = readRemovedAttackData;
areaAttacks.elementType = AreaAttackData;

beamAttacks.readKey = readAttackKey;
beamAttacks.readRemove = readRemovedAttackData;
beamAttacks.elementType = BeamAttackData;

droneAttacks.readKey
```

```
= readAttackKey;
droneAttacks.readRemove = readRemovedAttackData;
droneAttacks.elementType = DroneAttackData;

projectileAttacks.readKey = readAttackKey;
projectileAttacks.readRemove = readRemovedAttackData;
projectileAttacks.elementType = ProjectileAttackData;

participants.readKey = readPlayerKey;
participants.readRemove = null; // participants are never removed
participants.elementType = BattleParticipantInfo;
}

override public function read( input:BinaryInputStream ):void
{
hasBeenBaselined = true;

input.checkToken();


var activeSplit:String;
var numSplits:int = input.readUnsignedInt();
for (var i:int = 0; i < numSplits; ++i)
{
activeSplit = input.readUTF(); // split test prototype
if (activeSplit != '')
activeSplitPrototypes.push(activeSplit);
}

sector = ObjectPool.get(SectorData);
sector.id = input.readUTF();
sector.prototype = PrototypeModel.instance.getSectorPrototypeByName(input.readUTF());
sector.appearanceSeed = input.readInt();
maxSizeX = input.readInt();
maxSizeY = input.readInt();

galacticName = input.readUTF();
backgroundId = input.readInt();
planetId = input.readInt();
moonQuantity = input.readInt();
asteroidQuantity = input.readInt();
appearanceSeed = input.readInt64();

battleStartTick = input.readInt();
battleEndTick = input.readInt();
loadTimeoutTick = input.readInt();
battleState = input.readInt();
isBaseCombat = input.readBoolean();
isInstancedMission = input.readBoolean();
baseOwner = input.readUTF();
alloy
```

```
= input.readInt64();
synthetic = input.readInt64();
energy = input.readInt64();
credits = input.readInt64();
missionPersistence = input.readUTF();
players.read(input);
entities.read(input);
deadEntities.read(input);
areaAttacks.read(input);
beamAttacks.read(input);
droneAttacks.read(input);
input.checkToken();
projectileAttacks.read(input);
input.checkToken();

participants.read(input);
readAreaAttackHits(input);
readADMisses(input);
readADHits(input);

input.checkToken();
}

public function readVictors( input:BinaryInputStream ):Dictionary
{
var victors:Dictionary = new Dictionary();
var numVictors:int = input.readUnsignedInt();
for (var i:int = 0; i < numVictors; i++)
{
var key:String = input.readUTF();
victors[key] = true;
}
return victors;
}

public static var _propertyNames:Vector.<String> = new <String>[
"battleStartTick", // 0
"battleEndTick", // 1
"battleState", // 2
"players", // 3
"entities", // 4
"deadEntities", // 5
"areaAttacks", // 6
"beamAttacks", // 7
"droneAttacks", // 8
"projectileAttacks", // 9
"participants", // 10
"areaAttackHits", // 11
"adMisses", // 12
"adHits" // 13
];
```

```
public static var _propertyReaders:Vector.<String> = new <String>[
"readInt", // 0
"readInt", // 1
"readInt", // 2
null, // 3
null, // 4
null, // 5
null, // 6
null, // 7
null, // 8
null, // 9
null, // 10
"readAreaAttackHits", // 11
"readADMisses", // 12
"readADHits", // 13
];

override public function get propertyNames():Vector.<String>
{
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return _propertyReaders;
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in BattleBaselineResponse is not supported");
}


public function get battleState():int
{
return _battleState;
}

public function set battleState( newState:int ):void
{
_battleState = newState;
battleStateChanged = true;
}

public function get isTicked():Boolean { return false; }

public function get header():int
{
throw new Error("header in BattleData is not supported");
return
```

```
0;
}

public function set header( v:int ):void {}

public function get protocolID():int
{
throw new Error("protocolID in BattleData is not supported");
return 0;
}

public function set protocolID( v:int ):void {}

public function destroy():void
{
hasBeenBaselined = false;
if (sector)
ObjectPool.give(sector);
sector = null;

for (var i:int = 0; i < entities.length; i++)
ObjectPool.give(entities[i]);
entities.length = 0;

for (i = 0; i < deadEntities.length; i++)
ObjectPool.give(deadEntities[i]);
deadEntities.length = 0;

for (var id:String in adHits)
{
ObjectPool.give(adHits[id]);
delete adHits[id];
}
players.length = 0;
for (i = 0; i < areaAttacks.length; i++)
ObjectPool.give(areaAttacks[i]);
areaAttacks.length = 0;
for (i = 0; i < beamAttacks.length; i++)
ObjectPool.give(beamAttacks[i]);
beamAttacks.length = 0;
for (i = 0; i < droneAttacks.length; i++)
ObjectPool.give(droneAttacks[i]);
droneAttacks.length = 0;
for (i = 0; i < projectileAttacks.length; i++)
ObjectPool.give(projectileAttacks[i]);
projectileAttacks.length = 0;
for (i = 0; i < participants.length; i++)
ObjectPool.give(participants[i]);
participants.length = 0;
}
}
```

```
}


-----------------------------------
File 471: igw\com\service\server\incoming\data\BattleDebuff.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import com.service.server.replicable.ReplicableStruct;

public class BattleDebuff extends ReplicableStruct
{
public var prototype:String;
public var beginTick:int;
public var endTick:int;
public var stackCount:int;
public var magnitude:Number;

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
prototype = input.readUTF();
beginTick = input.readInt();
endTick = input.readInt();
stackCount = input.readInt();
magnitude = input.readFloat();
input.checkToken();
}

public static var _propertyNames:Vector.<String> = new <String>[
"beginTick", // 0
"endTick", // 1
"stackCount", // 2
"magnitude", // 3
];

public static var _propertyReaders:Vector.<String> = new <String>[
"readInt", // 0
"readInt", // 1
"readInt", // 2
"readFloat", // 3
];

override public function get propertyNames():Vector.<String>
{
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return
```

```
_propertyReaders;
}
}
}

-----------------------------------
File 472: igw\com\service\server\incoming\data\BattleEntityData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.replicable.ReplicableStruct;
import com.service.server.replicable.ReplicableVector;
import com.service.server.incoming.data.DebuffMapByType;

import flash.geom.Point;

import org.shared.ObjectPool;

public class BattleEntityData extends ReplicableStruct implements IServerData
{
public var id:String;
public var ownerId:String;
public var factionId:String;
public var type:int;

public var _location:Point = new Point();
public var gridLocationX:int;
public var gridLocationY:int;
public var targetLocation:Point = new Point();
public var _rotation:Number; // float
public var _velocity:Point = new Point();
public var currentHealth:int;
public var maxHealth:int;
public var organicTargetId:String;
public var radius:int;
public var selectedTargetId:String;
public var shieldsEnabled:Boolean;
public var shieldsCurrentHealth:int;
public var currentTargetId:String;
public var subsystemTarget:int;

public var shipPrototype:IPrototype;
public var buildingPrototype:IPrototype;
public var factionPrototype:String;

public var debuffs:DebuffMapByType = new DebuffMapByType;

private var _connectedPylons:Vector.<String>;
private
```

```
var _modules:Vector.<ModuleData>;
private var _subsystems:Vector.<SubsystemData>;
private var _weapons:ReplicableVector; // just the weapons portion of _modules
private var _activeDefenses:ReplicableVector; // just the active defenses portion of _modules

public function get rotation():Number
{
return _rotation;
}

public function set rotation( rot360:Number ):void
{
_rotation = rot360 * Math.PI / 180;
}

public function get velocity():Point
{
return _velocity;
}

public function set velocity( newVel:Point ):void
{
_velocity = newVel;
//trace( "set velocity for "+id+" to "+String(newVel) );
}

public function get location():Point
{
return _location;
}

public function set location( newVel:Point ):void
{
_location = newVel;
//trace( "set location for "+id+" to "+String(newVel) );
}

override public function decode( input:BinaryInputStream ):int
{
//trace( "decoding changes for "+id );
return super.decode(input);
}


public function BattleEntityData()
{
_connectedPylons = new Vector.<String>;
_modules = new Vector.<ModuleData>;
_weapons = new ReplicableVector;
_weapons.elementType = WeaponData;
_activeDefenses
```

```
= new ReplicableVector;
_activeDefenses.elementType = ActiveDefenseData;
_subsystems = new Vector.<SubsystemData>;
}

public override function read( input:BinaryInputStream ):void
{
var pm:PrototypeModel = PrototypeModel.instance;
input.checkToken();
id = input.readUTF();
ownerId = input.readUTF();
factionId = input.readUTF();
type = input.readInt();
location = readLocation(input);
gridLocationX = input.readInt();
gridLocationY = input.readInt();
targetLocation = readLocation(input);
//targetRotation = input.readDouble();
rotation = input.readFloat();
velocity = readLocation(input);
radius = input.readDouble();
currentHealth = input.readInt();
if (currentHealth < 0)
currentHealth = 0;
maxHealth = input.readInt();
shieldsEnabled = input.readBoolean();
shieldsCurrentHealth = input.readInt();
selectedTargetId = input.readUTF();
organicTargetId = input.readUTF();
subsystemTarget = input.readInt();
//executingPlayerMoveOrder = input.readBoolean();
shipPrototype = pm.getShipPrototype(input.readUTF());
buildingPrototype = pm.getBuildingPrototype(input.readUTF());
factionPrototype = input.readUTF();
//shipPersistence = input.readUTF();
//buildingPersistence = input.readUTF();
basicModules = input;
weaponsTemp = input;
activeDefensesTemp = input;
subsystems = input;

var numPylons:int = input.readUnsignedInt();
for (var i:int = 0; i < numPylons; i++)
{
_connectedPylons.push(input.readUTF());
}
//orbitingDrones
//minRange = input.readDouble();
//maxRange = input.readDouble();
debuffs.read(input);
input.checkToken();
```

```
}

public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

public static var _propertyNames:Vector.<String> = new <String>[
"location", // 0
"targetLocation", // 1
"rotation", // 2
"velocity", // 3
"radius", // 4
"currentHealth", // 5
"maxHealth", // 6
"shieldsEnabled", // 7
"selectedTargetId", // 8
"organicTargetId", // 9
"shieldsCurrentHealth", // 10
"weapons", // 11
"activeDefenses", // 12
"debuffs", // 13
];


public static var _propertyReaders:Vector.<String> = new <String>[
"readLocation", // 0
"readLocation", // 1
"readFloat", // 2
"readLocation", // 3
"readFloat", // 4
"readInt", // 5
"readInt", //6
"readBoolean", // 7
"readUTF", // 8
"readUTF", // 9
"readInt", // 10
];

override public function get propertyNames():Vector.<String>
{
```

```
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return _propertyReaders;
}

override public function resetDeltas():void
{
_weapons.resetDeltas();
_activeDefenses.resetDeltas();
debuffs.resetDeltas();
}

public function get activeDefenses():ReplicableVector { return _activeDefenses; }
public function set activeDefensesTemp( data:* ):void
{
var obj:IServerData;
if (data is BinaryInputStream)
{
var num:int = data.readUnsignedInt();
for (var i:int = 0; i < num; i++)
{
obj = ObjectPool.get(ActiveDefenseData);
obj.read(data);
_modules.push(obj);
_activeDefenses.push(obj);
}
} else
{
for (var key:String in data)
{
obj = ObjectPool.get(ActiveDefenseData);
obj.readJSON(data[key]);
_modules.push(obj);
_activeDefenses.push(obj);
}
}
}
public function set basicModules( data:* ):void
{
var obj:IServerData;
if (data is BinaryInputStream)
{
var num:int = data.readUnsignedInt();
for (var i:int = 0; i < num; i++)
{
obj = ObjectPool.get(ModuleData);
obj.read(data);
_modules.push(obj);
```

```
}
} else
{
for (var key:String in data)
{
obj = ObjectPool.get(ModuleData);
obj.readJSON(data[key]);
_modules.push(obj);
}
}
}
public function get connectedPylons():Vector.<String> { return _connectedPylons; }
public function get modules():Vector.<ModuleData> { return _modules; }

public function get weapons():ReplicableVector { return _weapons; }
public function set weaponsTemp( data:* ):void
{
var obj:IServerData;
if (data is BinaryInputStream)
{
var num:int = data.readUnsignedInt();
for (var i:int = 0; i < num; i++)
{
obj = ObjectPool.get(WeaponData);
obj.read(data);
_modules.push(obj);
_weapons.push(obj);
}
} else
{
for (var key:String in data)
{
obj = ObjectPool.get(WeaponData);
obj.readJSON(data[key]);
_modules.push(obj);
_weapons.push(obj);
}
}
}
public function get subsystemsList():Vector.<SubsystemData> { return _subsystems; }
public function set subsystems( data:* ):void
{
var obj:IServerData;
if (data is BinaryInputStream)
{
var num:int = data.readUnsignedInt();
for (var i:int = 0; i < num; i++)
{
obj = ObjectPool.get(SubsystemData);
obj.read(data);
```

```
_subsystems.push(obj);
}
} else
{
for (var key:String in data)
{
obj = ObjectPool.get(SubsystemData);
obj.readJSON(data[key]);
_subsystems.push(obj);
}
}
}

public function destroy():void
{
buildingPrototype = null;
shipPrototype = null;
for (var i:int = 0; i < _modules.length; i++)
{
ObjectPool.give(_modules[i]);
}
_modules.length = 0;
for (i = 0; i < _subsystems.length; i++)
{
ObjectPool.give(_subsystems[i]);
}
_subsystems.length = 0;
_connectedPylons.length = 0;
}
}
}

-------------------------------------
File 473: igw\com\service\server\incoming\data\BattleLogBaseDetailInfo.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class BattleLogBaseDetailInfo implements IServerData
{
public var baseHealth:Number;
public var buildings:Vector.<BattleLogEntityDetailInfo>;
public var baseRating:int;

public function read( input:BinaryInputStream ):void
{
buildings = new Vector.<BattleLogEntityDetailInfo>;
input.checkToken();
baseHealth = input.readDouble();
baseRating
```

```
= input.readInt();
input.checkToken();
}

public function readJSON( data:Object ):void
{
buildings = new Vector.<BattleLogEntityDetailInfo>;
baseRating = data.rating;
baseHealth = data.health;
}

public function destroy():void
{

}
}
}

-------------------------------------
File 474: igw\com\service\server\incoming\data\BattleLogEntityDetailInfo.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class BattleLogEntityDetailInfo implements IServerData
{
public var protoName:String;
public var health:Number;

public function read( input:BinaryInputStream ):void
{
var value:int = input.readUnsignedInt();
protoName = input.readUTF();
health = input.readDouble();
}

public function readJSON( data:Object ):void
{
protoName = data.prototype;
health = data.health;
}

public function destroy():void
{

}
}
}
-------------------------------------
File 475: igw\com\service\server\incoming\data\BattleLogFleetDetailInfo.as
package
```

```
com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class BattleLogFleetDetailInfo implements IServerData
{
public var name:String;
public var ships:Vector.<BattleLogEntityDetailInfo>;
public var fleetRating:int;
public var fleetHealth:Number;
public var alloyGained:int;
public var energyGained:int;
public var syntheticGained:int;
public var blueprintGained:String;

public function read( input:BinaryInputStream ):void
{
ships = new Vector.<BattleLogEntityDetailInfo>;
input.checkToken();
name = input.readUTF();
var shipsLength:int = input.readUnsignedInt();
var currentShip:BattleLogEntityDetailInfo;
for(var i:uint = 0; i < shipsLength; ++i)
{
currentShip = new BattleLogEntityDetailInfo();
currentShip.read(input);
ships.push(currentShip);
}
fleetRating = input.readInt();
fleetHealth = input.readDouble();
alloyGained = input.readInt();
energyGained = input.readInt();
syntheticGained = input.readInt();
input.checkToken();
}

public function readJSON( data:Object ):void
{
ships = new Vector.<BattleLogEntityDetailInfo>;
name = data.key;
var currentShip:BattleLogEntityDetailInfo;
for each( var shipJson:Object in data.shipResults )
{
currentShip = new BattleLogEntityDetailInfo();
currentShip.readJSON(shipJson);
ships.push(currentShip);
}

fleetRating = data.rating;
fleetHealth = data.health;
alloyGained
```

```
= data.alloyCargoChange;
energyGained = data.energyCargoChange;
syntheticGained = data.syntheticCargoChange;
}

public function destroy():void
{

}
}
}
```

File 476: igw\com\service\server\incoming\data\BattleLogPlayerDetailInfo.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import com.service.server.incoming.data.CrewInfo;
import org.shared.ObjectPool;

public class BattleLogPlayerDetailInfo implements IServerData
{
public var name:String;
public var playerKey:String;
public var race:String;
public var faction:String;
public var hasFleet:Boolean;
public var fleet:BattleLogFleetDetailInfo;
public var hasBase:Boolean;
public var base:BattleLogBaseDetailInfo;
public var level:int;
public var creditsGained:int;
public var blueprintGained:String;
public var crewGained:CrewInfo;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
name = input.readUTF();
playerKey = input.readUTF();
race = input.readUTF();
faction = input.readUTF();
hasFleet = input.readBoolean();
if (hasFleet)
{
fleet = new BattleLogFleetDetailInfo();
fleet.read(input);
}
hasBase = input.readBoolean();
if (hasBase)
{
base
```

```
= new BattleLogBaseDetailInfo();
base.read(input);
}
level = input.readInt();
creditsGained = input.readInt64();
blueprintGained = input.readUTF();
crewGained = ObjectPool.get(CrewInfo);
crewGained.read(input);
input.checkToken();
}

public function readJSON( data:Object ):void
{
name = data.name;
playerKey = data.playerId;
race = data.race;
faction = data.faction;
hasFleet = false;
if (data.fleetResult)
{
hasFleet = true;
fleet = new BattleLogFleetDetailInfo();
fleet.readJSON(data.fleetResult);
}
hasBase = false;
if (data.baseResult)
{
hasBase = true;
base = new BattleLogBaseDetailInfo();
base.readJSON(data.baseResult);
}
//level = input.level; // player level (not rating) seems to be unused
creditsGained = data.credits;
blueprintGained = data.blueprintGained;
if( data.crewGained )
{
crewGained = ObjectPool.get(CrewInfo);
crewGained.readJSON(data.crewGained);
}
}

public function destroy():void
{

}

}
}

-----------------------------------
File
```

477: igw\com\service\server\incoming\data\BattleLogPlayerSummaryInfo.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class BattleLogPlayerSummaryInfo implements IServerData
{
public var name:String;
public var playerKey:String;
public var race:String;
public var faction:String;
public var rating:int;
public var wasBase:Boolean;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
name = input.readUTF();
playerKey = input.readUTF();
race = input.readUTF();
faction = input.readUTF();
wasBase = input.readBoolean();
rating = input.readInt();
input.checkToken();
}

public function readJSON( data:Object ):void
{
name = data.name;
playerKey = data.playerId;
race = data.race;
faction = data.faction;
if( data["baseResult"] )
{
wasBase = true;
rating = data.baseResult.rating;
}
else
{
wasBase = false;
rating = data.fleetResult.rating;
}
}

public function destroy():void
{

}
}
}
```

File 478: igw\com\service\server\incoming\data\BattleLogSummaryInfo.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class BattleLogSummaryInfo implements IServerData
{

public var battleKey:String;
public var winners:Vector.<BattleLogPlayerSummaryInfo>;
public var losers:Vector.<BattleLogPlayerSummaryInfo>;
public var startTime:Number;
public var endTime:Number;
public var hasReplay:Boolean;

public function BattleLogSummaryInfo()
{
winners = new Vector.<BattleLogPlayerSummaryInfo>;
losers = new Vector.<BattleLogPlayerSummaryInfo>;
hasReplay = false;
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();
var i:uint;
var currentPlayer:BattleLogPlayerSummaryInfo;
battleKey = input.readUTF();
var winnersLength:int = input.readUnsignedInt();

for(; i < winnersLength; ++i)
{
currentPlayer = new BattleLogPlayerSummaryInfo();
currentPlayer.read(input);
winners.push(currentPlayer);
}

var losersLength:int = input.readUnsignedInt();

for(i = 0; i < losersLength; ++i)
{
currentPlayer = new BattleLogPlayerSummaryInfo();
currentPlayer.read(input);
losers.push(currentPlayer);
}

startTime = input.readInt64();
endTime = input.readInt64();
input.checkToken();
}
```

```
public function readJSON(data:Object):void
{
battleKey = data._id;
startTime = data.startTime;
endTime = data.endTime;
for each( var playerJson:Object in data.playerResults )
{
var playerSummary:BattleLogPlayerSummaryInfo = new BattleLogPlayerSummaryInfo();
playerSummary.readJSON( playerJson );
if( playerJson.victor == true )
{
winners.push(playerSummary);
}
else
{
losers.push(playerSummary);
}
}
hasReplay = data.hasReplay;
}

public function destroy():void
{
}
}
}
```

File 479: igw\com\service\server\incoming\data\BeamAttackData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import flash.geom.Point;

public class BeamAttackData extends AttackData
{
public var sourceAttachPoint:String;
public var targetAttachPoint:String;
public var targetScatterX:Number;
public var targetScatterY:Number;
public var maxRange:Number;
public var attackHit:Boolean;
public var hitLocation:Point;
public var hitTarget:String;

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
super.read(input);
sourceAttachPoint
```

```
= input.readUTF();
targetAttachPoint = input.readUTF();
targetScatterX = input.readDouble();
targetScatterY = input.readDouble();
maxRange = input.readDouble();
attackHit = input.readBoolean();
hitLocation = readLocation(input);
hitTarget = input.readUTF();
input.checkToken();
}

public static var _propertyNames:Vector.<String> = new <String>[
"targetEntityId", // 0
"rotation", // 1
"targetAttachPoint", // 2
"targetScatterX", // 3
"targetScatterY", // 4
"attackHit", // 5
"hitLocation", //6
"hitTarget", //7
];

public static var _propertyReaders:Vector.<String> = new <String>[
"readUTF", // 0
"readUnsignedByte", // 1
"readUTF", // 2
"readFloat", // 3
"readFloat", // 4
"readBoolean", // 5
"readLocation", //6
"readUTF", // 7
];

override public function get propertyNames():Vector.<String>
{
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return _propertyReaders;
}

override public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
```

```
// TODO - complain about missing key?
}
}
}

override public function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 480: igw\com\service\server\incoming\data\BlueprintData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.service.server.BinaryInputStream;

public class BlueprintData implements IServerData
{
public var id:String;
public var blueprintPrototype:String;
public var prototype:IPrototype;
public var playerOwner:String;
public var partsCollected:int;
public var partsCollectedBank:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

input.checkToken();
id = input.readUTF();
input.checkToken();

blueprintPrototype = input.readUTF();
playerOwner = input.readUTF();
partsCollected = input.readInt();
partsCollectedBank = input.readInt();

input.checkToken();
}

public function readJSON( data:Object ):void
{
id = data.key;
blueprintPrototype = data.blueprintPrototype;
playerOwner = data.playerOwner;
partsCollected
```

```
= data.partsCollected;
partsCollectedBank = data.partsCollectedBank;
}

public function destroy():void
{
prototype = null;
}
}
}

-----------------------------------
File 481: igw\com\service\server\incoming\data\BookmarksData.as
package com.service.server.incoming.data
{
import com.model.player.BookmarkVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

public class BookmarksData implements IServerData
{
public var bookmarks:Vector.<BookmarkVO>;

public function BookmarksData()
{
bookmarks = new Vector.<BookmarkVO>;
}

public function read( input:BinaryInputStream ):void
{
var prototypeModel:PrototypeModel = PrototypeModel.instance;

var bookmarksContainer:String = input.readUTF();
if (bookmarksContainer && bookmarksContainer != '')
{
var bookmarksBlob:Object = JSON.parse(bookmarksContainer);
var bookmark:BookmarkVO;
var sectorPrototypeName:String;
var sectorNamePrototypeName:String;
var sectorEnumPrototypeName:String;
var sectorPrototype:IPrototype;
var sectorNamePrototype:IPrototype;
var sectorEnumPrototype:IPrototype;
var index:uint;
for each (var currentBookmark:Object in bookmarksBlob.bookmarks)
{
sectorPrototypeName = (currentBookmark.hasOwnProperty('SectorProto')) ?
currentBookmark.SectorProto : '';
sectorNamePrototypeName = (currentBookmark.hasOwnProperty('NameProto')) ?
currentBookmark.NameProto
```

```
: '';
sectorEnumPrototypeName = (currentBookmark.hasOwnProperty('EnumProto')) ?
currentBookmark.EnumProto : '';

if (sectorPrototypeName != null && sectorPrototypeName != '')
sectorPrototype = prototypeModel.getSectorPrototypeByName(sectorPrototypeName);

if (sectorNamePrototypeName != null && sectorNamePrototypeName != '')
sectorNamePrototype =
prototypeModel.getSectorNamePrototypeByName(sectorNamePrototypeName);

if (sectorEnumPrototypeName != null && sectorEnumPrototype != '')
sectorEnumPrototype =
prototypeModel.getSectorNamePrototypeByName(sectorEnumPrototypeName);

bookmark = new BookmarkVO(currentBookmark.name, currentBookmark.sector,
sectorPrototype, sectorNamePrototype, sectorEnumPrototype, currentBookmark.x,
currentBookmark.y, index);
bookmarks.push(bookmark);
++index;
}
}
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in BookmarksData is not supported");
}

public function destroy():void
{
bookmarks.length = 0;
}
}
}

-------------------------------------
File 482: igw\com\service\server\incoming\data\BuffData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

public class BuffData implements IServerData
{
public var baseID:String;
public var began:Number;
public var ends:Number;
public var id:String;
public
```

```
var playerOwnerID:String;
public var prototype:IPrototype;
public var timeRemaining:Number;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
id = input.readUTF();
input.checkToken();

prototype = PrototypeModel.instance.getBuffPrototype(input.readUTF());
/* storeItemPrototype*/
input.readUTF();
baseID = input.readUTF();
playerOwnerID = input.readUTF();
began = input.readInt64();
ends = input.readInt64();

input.checkToken();
}

public function set now( v:Number ):void
{
timeRemaining = ends - v;
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in BuffData is not supported");
}

public function destroy():void
{
prototype = null;
}
}
}

-------------------------------------
File 483: igw\com\service\server\incoming\data\BuildingData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

import flash.utils.Dictionary;

public class BuildingData implements IServerData
{
```

```
public var baseID:String;
public var baseX:Number;
public var baseY:Number;
public var buildState:int;
public var currentHealth:Number;
public var id:String;
public var modules:Dictionary;
public var playerOwnerID:String;
public var refitModules:Dictionary;
public var prototype:IPrototype;
public var type:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

input.checkToken();
id = input.readUTF();
input.checkToken();

var prototypeModel:PrototypeModel = PrototypeModel.instance;
type = input.readUTF();
prototype = prototypeModel.getBuildingPrototype(type);
baseID = input.readUTF();
playerOwnerID = input.readUTF();

baseX = input.readInt();
baseY = input.readInt();

currentHealth = input.readDouble();
if (currentHealth < 0)
currentHealth = 0;
buildState = input.readInt();

// current modules
var slotName:String;
var modulePrototype:String;
var numModules:int = input.readUnsignedInt();
if (numModules > 0)
modules = new Dictionary();
for (var j:int = 0; j < numModules; j++)
{
slotName = input.readUTF();
modulePrototype = input.readUTF();
modules[slotName] = prototypeModel.getWeaponPrototype(modulePrototype);
}

// refit modules
numModules = input.readUnsignedInt();
if
```

```
(numModules > 0)
refitModules = new Dictionary();
for (j = 0; j < numModules; j++)
{
slotName = input.readUTF();
modulePrototype = input.readUTF();
refitModules[slotName] = prototypeModel.getWeaponPrototype(modulePrototype);
}

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in BuildingData is not supported");
}

public function destroy():void
{
modules = null;
prototype = null;
refitModules = null;
}
}
}

------------------------------------
File 484: igw\com\service\server\incoming\data\CrewInfo.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class CrewInfo implements IServerData
{
public var prototype:String;
public var rank:String;
public var rarity:int;
public var firstName:String; // first name prototype
public var lastName:String; // last name prototype

public function read( input:BinaryInputStream ):void
{
input.checkToken();
prototype = input.readUTF();
rank = input.readUTF();
rarity = input.readInt();
firstName = input.readUTF();
lastName = input.readUTF();
input.checkToken();
}

public
```

```
function readJSON( data:Object ):void
{
prototype = data.prototype;
rank = data.rank;
rarity = data.rarity;
firstName = data.firstName;
lastName = data.lastName;
}

public function destroy():void
{
}
}
}

-----------------------------------
File 485: igw\com\service\server\incoming\data\CrewMemberData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.service.server.BinaryInputStream;

public class CrewMemberData implements IServerData
{
public var fleetOwner:String;
public var crewMemberPrototype:String;
public var rarity:int;
public var rank:String;
public var playerOwner:String;
public var firstName:Number;
public var lastName:Number;
public var xp:Number;
public var crewState:int;
public var trainingStarted:Number;
public var trainingEnds:Number;
public var indoctrinated:Boolean;
public var pendingXp:Number;
public var pendingAdvancementSlot1:String;
public var pendingAdvancementSlot2:String;
public var pendingAdvancementSlot3:String;
public var pendingAdvancementSlot4:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

fleetOwner = input.readUTF();
crewMemberPrototype = input.readUTF();
rarity = input.readInt();
rank = input.readUTF();
playerOwner
```

```
= input.readUTF();
firstName = input.readInt64();
lastName = input.readInt64();
xp = input.readInt64();
crewState = input.readInt();
trainingStarted = input.readInt64();
trainingEnds = input.readInt64();
indoctrinated = input.readBoolean();
pendingXp = input.readInt64();
pendingAdvancementSlot1 = input.readUTF();
pendingAdvancementSlot2 = input.readUTF();
pendingAdvancementSlot3 = input.readUTF();
pendingAdvancementSlot4 = input.readUTF();

input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function destroy():void
{
}
}
}

-----------------------------------
File 486: igw\com\service\server\incoming\data\DebuffMapByType.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import com.service.server.replicable.ReplicableMap;
import com.service.server.incoming.data.RemovedObjectData;


import org.shared.ObjectPool;

public dynamic class DebuffMapByType extends ReplicableMap
{
public override function readKey( input:BinaryInputStream ):*
{
return input.readUTF();
}

public override function readRemove( input:BinaryInputStream ):*
{
var data:RemovedObjectData = ObjectPool.get(RemovedObjectData);
data.id = input.readUTF();
return data;
}
```

```
public override function createValue():*
{
return ObjectPool.get(DebuffMapByWeapon);
}
}
}
```

File 487: igw\com\service\server\incoming\data\DebuffMapByWeapon.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import com.service.server.replicable.ReplicableMap;
import com.service.server.incoming.data.BattleDebuff;


import org.shared.ObjectPool;

public dynamic class DebuffMapByWeapon extends ReplicableMap
{
public override function readKey( input:BinaryInputStream ):*
{
return input.readUTF();
}

public override function readRemove( input:BinaryInputStream ):*
{
var data:RemovedObjectData = ObjectPool.get(RemovedObjectData);
data.id = input.readUTF();
return data;
}
public override function createValue():*
{
return ObjectPool.get(BattleDebuff);
}
}
}
```

File 488: igw\com\service\server\incoming\data\DebugLineData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

import org.starling.utils.Color;

public class DebugLineData implements IServerData
{
public var id:uint;
public
```

```
var startX:Number;
public var startY:Number;
public var endX:Number;
public var endY:Number;
public var startColorR:uint;
public var startColorG:uint;
public var startColorB:uint;
public var endColorR:uint;
public var endColorG:uint;
public var endColorB:uint;
public var relativeToKey:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
id = input.readUnsignedInt();
startX = input.readDouble();
startY = input.readDouble();
endX = input.readDouble();
endY = input.readDouble();
startColorR = input.readUnsignedByte();
startColorG = input.readUnsignedByte();
startColorB = input.readUnsignedByte();
endColorR = input.readUnsignedByte();
endColorG = input.readUnsignedByte();
endColorB = input.readUnsignedByte();
relativeToKey = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function destroy():void
{
}

public function get startColor():uint
{
return Color.rgb(startColorR, startColorG, startColorB);
}

public function get endColor():uint
{
return Color.rgb(endColorR, endColorG, endColorB);
}
}
}

-----------------------------------
```

File 489: igw\com\service\server\incoming\data\DroneAttackData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class DroneAttackData extends AttackData
{
public var isOrbiting:Boolean;

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
super.read(input);
location = readLocation(input);
isOrbiting = input.readBoolean()

input.checkToken();
}

public static var _propertyNames:Vector.<String> = new <String>[
"location", // 0
"rotation", // 1
"targetEntityId", // 2
"isOrbiting", // 3
];

public static var _propertyReaders:Vector.<String> = new <String>[
"readLocation", // 0
"readUnsignedByte", // 1
"readUTF", // 2
"readBoolean", // 3
];

override public function get propertyNames():Vector.<String>
{
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return _propertyReaders;
}

override public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
}
```

```
else
{
// TODO - complain about missing key?
}
}
}

override public function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 490: igw\com\service\server\incoming\data\FleetData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class FleetData implements IServerData
{
public var starbaseID:String;
public var defendTarget:String;
public var currentCargo:Number;
public var cargoCapacity:Number;
public var loadSpeed:Number;
public var currentHealth:Number;
public var id:String;
public var level:int;
public var name:String;
public var ownerID:String;
public var sector:String;
public var sectorLocationX:Number;
public var sectorLocationY:Number;
public var ships:Vector.<String> = new Vector.<String>;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
id = input.readUTF();
input.checkToken();

ownerID = input.readUTF();
sector = input.readUTF();
starbaseID = input.readUTF();
name = input.readUTF();
sectorLocationX = input.readDouble();
sectorLocationY = input.readDouble();
currentHealth
```

```
= input.readDouble();
level = input.readInt();

currentCargo = 0;
currentCargo += input.readInt64(); // alloy
currentCargo += input.readInt64(); // energy
currentCargo += input.readInt64(); // synthetic
cargoCapacity = input.readInt64(); // cargo capacity
loadSpeed = input.readDouble();
defendTarget = input.readUTF();

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in FleetData is not supported");
}

public function destroy():void
{
ships.length = 0;
}
}
}

-----------------------------------
File 491: igw\com\service\server\incoming\data\IServerData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public interface IServerData
{
function read( input:BinaryInputStream ):void;
function readJSON( data:Object ):void;

function destroy():void;
}
}

-----------------------------------
File 492: igw\com\service\server\incoming\data\LeaderboardAllianceData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class LeaderboardAllianceData implements IServerData
{
public var key:String;
public
```

```
var name:String;
public var factionPrototype:String;
public var publicAlliance:Boolean;
public var numMembers:int;
public var totalCommendationPointsPvE:uint;
public var totalCommendationPointsPvP:uint;
public var totalWins:uint;
public var totalLosses:uint;

public var totalHighestFleetRating:Number;
public var avgHighestFleetRating:Number;
public var totalBlueprintPartsCollected:Number;

public var totalRating:Number;
public var avgRating:Number;
public var totalExperience:Number;

public var numMembersRank:int;
public var totalBaseRatingRank:int;
public var avgBaseRatingRank:int;
public var totalExperienceRank:int;
public var totalCommendiationCombinedRank:int;
public var totalWinsRank:int;
public var totalKdrRank:int;
public var totalHighestFleetRank:int;
public var avgHighestFleetRank:int;
public var totalBlueprintPartsRank:int;

public var totalqualifiedWinsPvP:Number;
public var totalBubbleHoursGranted:Number;
public var totalcurrentPVPEvent:Number;
public var totalcurrentPVPEventQuarter:Number;
public var totalCreditsTradeRoute:Number;
public var totalResourcesTradeRoute:Number;
public var totalResourcesSalvaged:Number;
public var totalCreditsBounty:Number;
public var totalWinsVsBase:Number;

public var totalqualifiedWinsPvPRank:int;
public var totalBubbleHoursGrantedRank:int;
public var totalcurrentPVPEventRank:int;
public var totalcurrentPVPEventQuarterRank:int;
public var totalCreditsTradeRouteRank:int;
public var totalResourcesTradeRouteRank:int;
public var totalResourcesSalvagedRank:int;
public var totalCreditsBountyRank:int;
public var totalWinsVsBaseRank:int;

public function read(input:BinaryInputStream):void
{
input.checkToken();
```

```
input.checkToken();
key = input.readUTF(); // playerKey
input.checkToken();
name = input.readUTF(); // name
factionPrototype = input.readUTF();
publicAlliance = input.readBoolean();
numMembers = input.readInt();
totalRating = input.readInt();
avgRating = input.readDouble();
totalExperience = input.readInt64();
totalCommendationPointsPvE = input.readInt64();
totalCommendationPointsPvP = input.readInt64();
totalWins = input.readInt64();
totalLosses = input.readInt64();
totalHighestFleetRating = input.readInt64();
avgHighestFleetRating = input.readDouble();
totalBlueprintPartsCollected = input.readInt64();

totalqualifiedWinsPvP = input.readInt64();
totalBubbleHoursGranted = input.readInt64();
totalcurrentPVPEvent = input.readInt64();
totalcurrentPVPEventQuarter = input.readInt64();
totalCreditsTradeRoute = input.readInt64();
totalResourcesTradeRoute = input.readInt64();
totalResourcesSalvaged = input.readInt64();
totalCreditsBounty = input.readInt64();
totalWinsVsBase = input.readInt64();

numMembersRank = input.readInt();
totalBaseRatingRank = input.readInt();
avgBaseRatingRank = input.readInt();
totalExperienceRank = input.readInt();
totalCommendiationCombinedRank = input.readInt();
totalWinsRank = input.readInt();
totalKdrRank = input.readInt();
totalHighestFleetRank = input.readInt();
avgHighestFleetRank = input.readInt();
totalBlueprintPartsRank = input.readInt();

totalqualifiedWinsPvPRank = input.readInt();
totalBubbleHoursGrantedRank = input.readInt();
totalcurrentPVPEventRank = input.readInt();
totalcurrentPVPEventQuarterRank = input.readInt();
totalCreditsTradeRouteRank = input.readInt();
totalResourcesTradeRouteRank = input.readInt();
totalResourcesSalvagedRank = input.readInt();
totalCreditsBountyRank = input.readInt();
totalWinsVsBaseRank = input.readInt();

input.checkToken();
```

```
}

public function readJSON(data:Object):void
{
throw new Error("readJSON is not supported");
}

public function destroy():void
{
}
}
}
```

File 493: igw\com\service\server\incoming\data\LeaderboardData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

import org.shared.ObjectPool;

public class LeaderboardData implements IServerData
{
public var leaderboardType:int;
public var leaderboardScope:int;
public var players:Vector.<LeaderboardPlayerData>;
public var alliances:Vector.<LeaderboardAllianceData>;

public function LeaderboardData()
{
players = new Vector.<LeaderboardPlayerData>;
alliances = new Vector.<LeaderboardAllianceData>;
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();

leaderboardType = input.readInt(); // leaderboard type
leaderboardScope = input.readInt();

var num:int = input.readUnsignedInt();
var currentLeaderboardPlayer:LeaderboardPlayerData;
for (var idx:int = 0; idx < num; ++idx)
{
currentLeaderboardPlayer = ObjectPool.get(LeaderboardPlayerData);
currentLeaderboardPlayer.read(input);
players.push(currentLeaderboardPlayer);
}

var
```

```
numAlliances:int = input.readUnsignedInt();
var currentLeaderboardAlliance:LeaderboardAllianceData;
for (idx = 0; idx < numAlliances; ++idx)
{
currentLeaderboardAlliance = ObjectPool.get(LeaderboardAllianceData);
currentLeaderboardAlliance.read(input);
alliances.push(currentLeaderboardAlliance);
}

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function destroy():void
{
players.length = 0;
alliances.length = 0;
}
}
}

-----------------------------------
File 494: igw\com\service\server\incoming\data\LeaderboardPlayerData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class LeaderboardPlayerData implements IServerData
{
public var playerKey:String;
public var name:String;
public var racePrototype:String;
public var experience:uint;
public var commendationPointsPvE:uint;
public var commendationPointsPvP:uint;
public var wins:uint;
public var losses:uint;
public var draws:uint;
public var allianceKey:String;
public var allianceName:String;
public var sectorOwner:String;
public var baseRating:Number;
public var highestFleetRating:int;
public var blueprintPartsCollected:int;
public var baseRatingRank:int;
public var experienceRank:int;
public
```

```
var commendiationCombinedRank:int;
public var winsRank:int;
public var kdrRank:int;
public var highestFleetRank:int;
public var blueprintPartsRank:int;

public var qualifiedWinsPvP:Number;
public var BubbleHoursGranted:Number;
public var currentPVPEvent:Number;
public var currentPVPEventQuarter:Number;
public var CreditsTradeRoute:Number;
public var ResourcesTradeRoute:Number;
public var ResourcesSalvaged:Number;
public var CreditsBounty:Number;
public var WinsVsBase:Number;

public var qualifiedWinsPvPRank:int;
public var BubbleHoursGrantedRank:int;
public var currentPVPEventRank:int;
public var currentPVPEventQuarterRank:int;
public var CreditsTradeRouteRank:int;
public var ResourcesTradeRouteRank:int;
public var ResourcesSalvagedRank:int;
public var CreditsBountyRank:int;
public var WinsVsBaseRank:int;

public function read(input:BinaryInputStream):void
{
input.checkToken();
input.checkToken();
playerKey = input.readUTF(); // playerKey
input.checkToken();
name = input.readUTF(); // name



racePrototype = input.readUTF(); // racePrototype
experience = input.readInt64(); // experience
commendationPointsPvE = input.readInt64(); // commendiation points pve
commendationPointsPvP = input.readInt64(); // commendiation points pvp
wins = input.readInt64(); // wins
losses = input.readInt64(); // losses
draws = input.readInt64(); // draws
qualifiedWinsPvP = input.readInt64();
BubbleHoursGranted = input.readInt64();
currentPVPEvent = input.readInt64();
currentPVPEventQuarter = input.readInt64();
CreditsTradeRoute = input.readInt64();
ResourcesTradeRoute = input.readInt64();
ResourcesSalvaged = input.readInt64();
CreditsBounty
```

```
= input.readInt64();
WinsVsBase = input.readInt64();

allianceKey = input.readUTF(); // alliance key
allianceName = input.readUTF(); // alliance name
sectorOwner = input.readUTF();
baseRating = input.readInt(); // baseRating
highestFleetRating = input.readInt();
blueprintPartsCollected = input.readInt();
baseRatingRank = input.readInt();
experienceRank = input.readInt();
commendiationCombinedRank = input.readInt();
winsRank = input.readInt();
kdrRank = input.readInt();
highestFleetRank = input.readInt();
blueprintPartsRank = input.readInt();

qualifiedWinsPvPRank = input.readInt();
BubbleHoursGrantedRank = input.readInt();
currentPVPEventRank = input.readInt();
currentPVPEventQuarterRank = input.readInt();
CreditsTradeRouteRank = input.readInt();
ResourcesTradeRouteRank = input.readInt();
ResourcesSalvagedRank = input.readInt();
CreditsBountyRank = input.readInt();
WinsVsBaseRank = input.readInt();

input.checkToken();
}

public function readJSON(data:Object):void
{
throw new Error("readJSON is not supported");
}

public function destroy():void
{
}
}
}
---------------------------------
File 495: igw\com\service\server\incoming\data\MailInboxData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class MailInboxData implements IServerData
{

public var key:String;
public
```

```
var sender:String;
public var subject:String;
public var isRead:Boolean;
public var timeSent:Number;


public function read(input:BinaryInputStream):void
{
input.checkToken();
key = input.readUTF(); // key
sender = input.readUTF(); // senderName
subject = input.readUTF(); // subject
isRead = input.readBoolean(); // read flag
timeSent = input.readInt64(); // when it was sent (UTC millis)
input.checkToken();

}

public function readJSON(data:Object):void
{
key = data.key;
sender = data.senderName;
subject = data.subject;
isRead = data.readFlag;
timeSent = data.timeSent;

}

public function destroy():void
{
}
}
}
```

File 496: igw\com\service\server\incoming\data\MissionData.as

```
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.service.server.BinaryInputStream;

public class MissionData implements IServerData
{
public var accepted:Boolean;
public var id:String;
public var missionPrototype:String;
public var prototype:IPrototype;
public var playerOwner:String;
public var progress:int;
public var rewardAccepted:Boolean;
public var sector:String;
public
```

```
var transgate:String;
public var transGateSectorLocationX:Number;
public var transGateSectorLocationY:Number;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

input.checkToken();
id = input.readUTF();
input.checkToken();

missionPrototype = input.readUTF();
playerOwner = input.readUTF();
transgate = input.readUTF();
sector = input.readUTF();

progress = input.readInt();
accepted = input.readBoolean();
rewardAccepted = input.readBoolean();

input.readUTF(); //accepting fleet

transGateSectorLocationX = input.readDouble();
transGateSectorLocationY = input.readDouble();


input.checkToken();
}

public function readJSON( data:Object ):void
{
accepted = data.accepted;
id = data.key;
missionPrototype = data.missionPrototype;
playerOwner = data.playerOwner;
progress = data.progress;
rewardAccepted = data.rewardAccepted;
sector = data.sector;
transgate = data.transGate;
}

public function get category():String { return prototype.getUnsafeValue("category"); }

public function destroy():void
{
prototype = null;
}
}
}

-----------------------------------
```

File 497: igw\com\service\server\incoming\data\MissionScoreData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class MissionScoreData implements IServerData
{
public var key:String;
public var instancedMissionID:int;
public var bestTime:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
key = input.readUTF();
input.checkToken();
instancedMissionID = input.readInt();
bestTime = input.readInt();

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in MissionScoreData is not supported");
}

public function destroy():void
{
}
}
}
```

File 498: igw\com\service\server\incoming\data\ModuleData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;
import com.service.server.replicable.ReplicableStruct;

public class ModuleData extends ReplicableStruct implements IServerData
{
public var moduleIdx:int;
public var type:int;
public var isActive:Boolean;

public var offsetX:int;
public var offsetY:int;

public
```

```
var modulePrototype:String;
public var weaponPrototype:String;
public var activeDefensePrototype:String;
public var slotPrototype:String;
public var attachPointPrototype:String;

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
moduleIdx = input.readInt();
type = input.readInt();
isActive = input.readBoolean();
offsetX = input.readDouble();
offsetY = input.readDouble();
modulePrototype = input.readUTF();
weaponPrototype = input.readUTF();
activeDefensePrototype = input.readUTF();
slotPrototype = input.readUTF();
attachPointPrototype = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

public function destroy():void
{

}
}
}



---
File 499: igw\com\service\server\incoming\data\MotdData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public
```

```
class MotdData implements IServerData
{
public var key:String;
public var startTime:Number;
public var title:String;
public var subtitle:String;
public var text:String;
public var imageURL:String;
public var isRead:Boolean;


public function read(input:BinaryInputStream):void
{
input.checkToken();
input.checkToken();
key = input.readUTF();
input.checkToken();
startTime = input.readInt64();
title = input.readUTF();
subtitle = input.readUTF();
text = input.readUTF();
imageURL = input.readUTF();
input.checkToken();

isRead = input.readBoolean();
}

public function readJSON(data:Object):void
{
// Unimplemented.
}

public function destroy():void
{
}
}
}
```

File 500: igw\com\service\server\incoming\data\ProjectileAttackData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

import flash.geom.Point;

public class ProjectileAttackData extends AttackData
{
public var end:Point = new Point();
public var finishTick:int;
public var guided:Boolean;

public
```

```
var fadeTime:Number = -1;

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
super.read(input);
location = readLocation(input);
end = readLocation(input);
finishTick = input.readInt();
guided = input.readBoolean();
input.checkToken();
}

public static var _propertyNames:Vector.<String> = new <String>[
"location", // 0
"rotation", // 1
"targetEntityId", // 2
];

public static var _propertyReaders:Vector.<String> = new <String>[
"readLocation", // 0
"readUnsignedByte", // 1
"readUTF", // 2
];

override public function get propertyNames():Vector.<String>
{
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return _propertyReaders;
}

override public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
}
}
}

override public function destroy():void
{

}
}
}
```

File 501: igw\com\service\server\incoming\data\RemovedAttackData.as
```
package com.service.server.incoming.data
{
import com.enum.RemoveReasonEnum;
import com.service.server.BinaryInputStream;

public class RemovedAttackData implements IServerData
{
public var id:int;
public var reason:int;
public var x:Number;
public var y:Number;

public function read( input:BinaryInputStream ):void
{
id = input.readUnsignedInt();
reason = input.readInt();
if (reason == RemoveReasonEnum.AttackComplete || reason ==
RemoveReasonEnum.ShieldComplete || reason == RemoveReasonEnum.Intercepted)
{
x = input.readDouble();
y = input.readDouble();
}
}

public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

public function destroy():void
{

}
}
}
```

File 502: igw\com\service\server\incoming\data\RemovedObjectData.as
```
package
```

```
com.service.server.incoming.data
{
import com.enum.RemoveReasonEnum;
import com.service.server.BinaryInputStream;

public class RemovedObjectData implements IServerData
{
public var id:String;
public var reason:int;
public var x:Number;
public var y:Number;

public function read( input:BinaryInputStream ):void
{
id = input.readUTF();
reason = input.readInt();
if (reason == RemoveReasonEnum.AttackComplete || reason ==
RemoveReasonEnum.ShieldComplete || reason == RemoveReasonEnum.Intercepted)
{
x = input.readDouble();
y = input.readDouble();
}
}

public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

public function destroy():void
{

}
}
}

-----------------------------------
File 503: igw\com\service\server\incoming\data\ResearchData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import
```

```
com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

public class ResearchData implements IServerData
{
public var baseID:String;
public var id:String = '';
public var playerOwnerID:String;
public var prototype:IPrototype;
public var buildState:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
id= input.readUTF();
input.checkToken();

prototype = PrototypeModel.instance.getResearchPrototypeByName(input.readUTF());
baseID = id.substr(0,id.indexOf(".research"));
//input.readUTF();
playerOwnerID = id.substr(0,id.indexOf(".base"));
//input.readUTF();
//input.readUTF(); //building owner id
buildState = input.readInt();

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in ResearchData is not supported");
}

public function destroy():void
{
prototype = null;
}
}
}

-----------------------------------
File 504: igw\com\service\server\incoming\data\ScoreData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class ScoreData implements IServerData
{
public var key:String;
public
```

```
var scoreKey:String;
public var value:Number;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
key = input.readUTF();
input.checkToken();
scoreKey = input.readUTF();
value = input.readInt64();

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in ScoreData is not supported");
}

public function destroy():void
{
}
}
}

-------------------------------------
File 505: igw\com\service\server\incoming\data\SectorBattleData.as
package com.service.server.incoming.data
{
import com.enum.FactionEnum;
import com.model.player.PlayerModel;
import com.model.player.PlayerVO;
import com.service.server.BinaryInputStream;

public class SectorBattleData implements IServerData
{
public var id:String;
public var locationX:Number;
public var locationY:Number;
public var mapSizeX:int;
public var mapSizeY:int;
public var participantPlayers:Array = [];
public var participantFleets:Array = [];
public var participantBase:String;
public var observerIds:Array = [];
public var serverIdentifier:String;
public var joinable:Boolean;
public var maxPlayersPerFaction:int;
public var maxFactions:int;
public
```

```
var pvp:int;
public var firstJoiningFactionPrototype:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
id = input.readUTF();
locationX = input.readDouble();
locationY = input.readDouble();

mapSizeX = input.readInt();
mapSizeY = input.readInt();

var numParticipants:int = input.readUnsignedInt();
var playerId:String;
var currentUser:PlayerVO;
for (var i:int = 0; i < numParticipants; i++)
{
playerId = input.readUTF();
participantPlayers.push(playerId);
}

numParticipants = input.readUnsignedInt();
for (i = 0; i < numParticipants; i++)
{
participantFleets.push(input.readUTF());
}

participantBase = input.readUTF();

var numObservers:int = input.readUnsignedInt();
for (i = 0; i < numObservers; i++)
{
observerIds.push(input.readUTF());
}
serverIdentifier = input.readUTF();
joinable = input.readBoolean();
maxPlayersPerFaction = input.readUnsignedShort();
maxFactions = input.readUnsignedShort();
input.readUTF();
pvp = input.readInt64();
firstJoiningFactionPrototype = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
```

```
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}
public function destroy():void
{
participantPlayers.length = 0;
participantFleets.length = 0;
observerIds.length = 0;
}
}
}

-----------------------------------
File 506: igw\com\service\server\incoming\data\SectorData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

public class SectorData implements IServerData
{
public var appearanceSeed:int;
public var id:String;
public var neighborhood:Number;
public var prototype:IPrototype;
public var sectorEnum:IPrototype;
public var sectorName:IPrototype;
public var splitTestCohortPrototype:IPrototype;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
id = input.readUTF();
input.checkToken();

prototype = PrototypeModel.instance.getSectorPrototypeByName(input.readUTF());
sectorName = PrototypeModel.instance.getSectorNamePrototypeByName(input.readUTF());
sectorEnum = PrototypeModel.instance.getSectorNamePrototypeByName(input.readUTF());
neighborhood = input.readInt64();
appearanceSeed = input.readInt();
splitTestCohortPrototype =
PrototypeModel.instance.getSplitTestCohortPrototypeByName(input.readUTF());

input.checkToken();
```

```
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in ShipData is not supported");
}

public function destroy():void
{
prototype = null;
sectorEnum = null;
sectorName = null;
}
}
}

-----------------------------------
File 507: igw\com\service\server\incoming\data\SectorEntityData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class SectorEntityData implements IServerData
{
public var id:String;
public var ownerId:String;
public var name:String;
public var type:int;
public var state:int;
public var locationX:Number;
public var locationY:Number;
public var travelSpeed:int = 10;
public var currentHealthPct:Number;
public var shipPrototype:String;
public var factionPrototype:String;
public var bubbled:Boolean;
public var mission:Boolean;
public var isPositiveWarp:Boolean;
public var level:Number;
public var cargo:Number;
public var cloaked:Boolean;
public var eventSpawn:Boolean;
public var alertSector:Boolean;
public var baseRatingTech:Number;
public var maxPlayersPerFaction:Number;
public var additionalInfo:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
```

```
id = input.readUTF();
ownerId = input.readUTF();
name = input.readUTF();
type = input.readInt();
state = input.readInt();
locationX = input.readDouble();
locationY = input.readDouble();
travelSpeed = input.readDouble();
currentHealthPct = input.readDouble();
shipPrototype = input.readUTF();
factionPrototype = input.readUTF();
bubbled = input.readBoolean();
mission = input.readBoolean();
isPositiveWarp = input.readBoolean();
level = input.readInt64();
cargo = input.readInt64();
cloaked = input.readBoolean();
eventSpawn = input.readBoolean();
alertSector = input.readBoolean();
baseRatingTech = input.readInt64();
maxPlayersPerFaction = input.readInt();
additionalInfo = input.readUTF();
var numChargeups:int = input.readUnsignedInt();
for (var i:int = 0; i < numChargeups; i++)
{
var orderType:int = input.readInt();
input.checkToken();
var startTick:int = input.readInt();
var endTick:int = input.readInt();
input.checkToken();
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function destroy():void
{
}
}
}

---
File 508: igw\com\service\server\incoming\data\SectorEntityUpdateData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public
```

```
class SectorEntityUpdateData implements IServerData
{
public var id:String;
public var bubbled:Boolean;
public var cargo:Number;
public var cloaked:Boolean;
public var currentHealthPct:Number;
public var rating:Number;
public var baseRatingTech:Number;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
bubbled = input.readBoolean();
cargo = input.readInt64();
cloaked = input.readBoolean();
currentHealthPct = input.readDouble();

var numChargeups:int = input.readUnsignedInt();
for (var i:int = 0; i < numChargeups; i++)
{
var orderType:int = input.readInt();
var startTick:int = input.readInt();
var endTick:int = input.readInt();
}

rating = input.readInt64();
baseRatingTech = input.readInt64();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function destroy():void
{

}
}
}

-----------------------------------
File 509: igw\com\service\server\incoming\data\SectorObjectiveData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class SectorObjectiveData implements IServerData
{
public
```

```
var missionKey:String;
public var locationX:Number;
public var locationY:Number;
public var asset:String;
public var type:String;

public function read(input:BinaryInputStream):void
{
missionKey = input.readUTF(); //Mission key
input.checkToken();
locationX = input.readDouble(); //locationX
locationY = input.readDouble(); //locationY
asset = input.readUTF(); //Asset
type = input.readUTF(); //Type
input.checkToken();
}

public function readJSON(data:Object):void
{
}

public function destroy():void
{
}
}
}
```

File 510: igw\com\service\server\incoming\data\SectorOrderData.as

```
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class SectorOrderData implements IServerData
{
public var id:String;
public var entityId:String;
public var orderType:int;
public var issuedTick:int;
public var finishTick:int;
public var originLocationX:Number;
public var originLocationY:Number;
public var targetLocationX:Number;
public var targetLocationY:Number;
public var targetId:String;
public var destinationSector:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
id = input.readUTF();
entityId
```

```
= input.readUTF();
orderType = input.readInt();
issuedTick = input.readInt();
finishTick = input.readInt();
originLocationX = input.readDouble();
originLocationY = input.readDouble();
targetLocationX = input.readDouble();
targetLocationY = input.readDouble();
targetId = input.readUTF();
destinationSector = input.readUTF();
input.checkToken();
}


public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

public function destroy():void
{

}
}
}

-------------------------------------
File 511: igw\com\service\server\incoming\data\ShipData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

import flash.utils.Dictionary;

public class ShipData implements IServerData
{
public var currentHealth:Number;
public var fleetOwner:String;
public var shipName:String;
public
```

```
var positionIndex:int;
public var i:int;
public var id:String;
public var modules:Dictionary;
public var ownerID:String;
public var prototype:IPrototype;
public var refitModules:Dictionary;
public var buildState:int;

public function read( input:BinaryInputStream ):void
{
var prototypeModel:PrototypeModel = PrototypeModel.instance;

input.checkToken();
input.checkToken();
id = input.readUTF();
input.checkToken();

prototype = prototypeModel.getShipPrototype(input.readUTF());
ownerID = input.readUTF();
fleetOwner = input.readUTF();
positionIndex = input.readInt();
//shipName = "Unknown";
shipName = input.readUTF();
/* refitShipName */
input.readUTF();
/* baseOwner = */
input.readUTF();
currentHealth = input.readDouble();
buildState = input.readInt();

var slotName:String;
var modulePrototype:String;

// current modules
var numModules:int = input.readUnsignedInt();
modules = new Dictionary();
for (i = 0; i < numModules; i++)
{
slotName = input.readUTF();
modulePrototype = input.readUTF();
modules[slotName] = prototypeModel.getWeaponPrototype(modulePrototype);
}

// refit modules
numModules = input.readUnsignedInt();
if (numModules > 0)
refitModules = new Dictionary();
for (i = 0; i < numModules; i++)
{
slotName
```

```
= input.readUTF();
modulePrototype = input.readUTF();
refitModules[slotName] = prototypeModel.getWeaponPrototype(modulePrototype);
}

//fill in any missing slots
var slots:Array = prototype.getValue("slots");
for (i = 0; i < slots.length; i++)
{
slotName = slots[i];
modules[slotName] = modules[slotName];
}

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in ShipData is not supported");
}

public function destroy():void
{
modules = null;
prototype = null;
refitModules = null;
}
}
}

-------------------------------------
File 512: igw\com\service\server\incoming\data\SubsystemData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class SubsystemData implements IServerData
{
//public var id:int;
public var subsystemPrototype:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
subsystemPrototype = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
for
```

```
(var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

public function destroy():void
{

}
}
}

------------------------------------
File 513: igw\com\service\server\incoming\data\TradeRouteData.as
package com.service.server.incoming.data
{
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

public class TradeRouteData implements IServerData
{
public var began:uint;
public var baseID:String;
public var bribe:Number;
public var contractPrototype:IPrototype;
public var duration:Number;
public var end:uint;
public var frequency:Number;
public var id:String;
public var name:String;
public var offerExpires:uint;
public var offerRejected:Boolean;
public var payout:Number;
public var productivity:Number;
public var reputation:Number;
public var reputationWhenBegun:Number;
public var security:Number;

private var _factionPrototype:IPrototype;

public
```

```
function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
id = input.readUTF();
input.checkToken();

var protoModel:PrototypeModel = PrototypeModel.instance;
var assetModel:AssetModel = AssetModel.instance;

factionPrototype = protoModel.getFactionPrototypeByName(input.readUTF());
baseID = input.readUTF();
reputation = input.readInt64();

offerExpires = input.readInt64();
contractPrototype = protoModel.getContractPrototypeByName(input.readUTF());
productivity = input.readDouble();
payout = input.readDouble();
duration = input.readDouble();
frequency = input.readDouble();
security = input.readDouble();
reputationWhenBegun = input.readInt64();
began = input.readInt64();
end = input.readInt64();
/* buildState */
input.readInt();

bribe = input.readDouble();
// extension
input.readDouble();
// efficiency
input.readDouble();

offerRejected = input.readBoolean();

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in TradeRouteData is not supported");
}

public function get corporation():String { return _factionPrototype.name; }

public function get factionPrototype():IPrototype { return _factionPrototype; }
public function set factionPrototype( v:IPrototype ):void
{
_factionPrototype = v;
var currentAssetVO:AssetVO =
AssetModel.instance.getEntityData(factionPrototype.getValue('uiAsset'));
```

```
name = currentAssetVO.visibleName;
}

public function destroy():void
{
contractPrototype = null;
factionPrototype = null;
}
}
}

-----------------------------------
File 514: igw\com\service\server\incoming\data\WarfrontData.as
package com.service.server.incoming.data
{
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;

public class WarfrontData
{
public var id:String;
public var battleServerAddress:String;

public var attackerID:String;
public var attackerName:String;
public var attackerRace:IPrototype;
public var attackerFleetRating:int;

public var defenderID:String;
public var defenderName:String;
public var defenderRace:IPrototype;
public var defenderFleetRating:int;
public var defenderBaseRating:int;

public var sector:String;
public var sectorX:Number;
public var sectorY:Number;

public function read( input:BinaryInputStream ):void
{
var prototypeModel:PrototypeModel = PrototypeModel.instance;

// map value
input.checkToken();
id = input.readUTF(); // battleKey
battleServerAddress = input.readUTF(); // battleIdentifier

attackerID = input.readUTF(); // attacker key
attackerName
```

```
= input.readUTF(); // attacker name
attackerRace = prototypeModel.getRacePrototypeByName(input.readUTF()); // attacker
racePrototype
attackerFleetRating = input.readInt(); // attacker fleet rating

defenderID = input.readUTF(); // defender key
defenderName = input.readUTF(); // defender name
defenderRace = prototypeModel.getRacePrototypeByName(input.readUTF()); // defender
racePrototype
defenderFleetRating = input.readInt(); // defender fleet rating
defenderBaseRating = input.readInt(); // defender baseRating

sector = input.readUTF(); // sector
sectorX = input.readDouble(); // sectorX
sectorY = input.readDouble(); // sectorY

input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in TradeRouteData is not supported");
}

public function destroy():void
{

}
}
}

-----------------------------------
File 515: igw\com\service\server\incoming\data\WeaponData.as
package com.service.server.incoming.data
{
import com.service.server.BinaryInputStream;

public class WeaponData extends ModuleData
{
public var weaponState:uint;
public var rotation:Number;

override public function read( input:BinaryInputStream ):void
{
input.checkToken();
super.read(input);
weaponState = input.readInt();
rotation = input.readUnsignedByte();
input.checkToken();
}

public
```

```
static var _propertyNames:Vector.<String>= new <String>[
"weaponState", // 0
"rotation", // 1
];


public static var _propertyReaders:Vector.<String>= new <String>[
"readInt", // 0
"readUnsignedByte", // 1
];

override public function get propertyNames():Vector.<String>
{
return _propertyNames;
}
override public function get propertyReaders():Vector.<String>
{
return _propertyReaders;
}


override public function readJSON( data:Object ):void
{
for (var key:String in data)
{
if (this.hasOwnProperty(key))
{
this[key] = data[key];
} else
{
// TODO - complain about missing key?
}
}
}

override public function destroy():void
{

}
}
}



-----------------------------------
File 516: igw\com\service\server\incoming\leaderboard\LeaderboardResponse.as
package com.service.server.incoming.leaderboard
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import
```

```
com.service.server.incoming.data.LeaderboardData;

import org.shared.ObjectPool;

public class LeaderboardResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var leaderboardData:LeaderboardData;
public function read( input:BinaryInputStream ):void
{
leaderboardData = ObjectPool.get(LeaderboardData);
leaderboardData.read(input);
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
leaderboardData.destroy();
leaderboardData = null;
}
}
}

-----------------------------------
File 517: igw\com\service\server\incoming\leaderboard\PlayerProfileResponse.as
package com.service.server.incoming.leaderboard
{
import com.model.player.PlayerVO;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class PlayerProfileResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public
```

```
var players:Vector.<PlayerVO>;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
var currentPlayer:PlayerVO;
players = new Vector.<PlayerVO>;
var num:int = input.readUnsignedInt();
for (var idx:int = 0; idx < num; ++idx)
{
currentPlayer = new PlayerVO();
input.checkToken();
currentPlayer.id = input.readUTF(); // playerKey
currentPlayer.name = input.readUTF(); // name
currentPlayer.avatarName = input.readUTF(); // racePrototype
currentPlayer.faction = input.readUTF(); //factionPrototype
currentPlayer.xp = input.readInt64(); // experience
currentPlayer.commendationPointsPVE = input.readInt64(); // commendiation points pve
currentPlayer.commendationPointsPVP = input.readInt64(); // commendiation points pvp
currentPlayer.wins = input.readInt64(); // wins
currentPlayer.losses = input.readInt64(); // losses
currentPlayer.draws = input.readInt64(); // draws
currentPlayer.lastOnline = input.readInt64(); // last online utc millis
currentPlayer.baseRating = input.readInt(); // baseRating
currentPlayer.baseSector = input.readUTF(); // baseSector
currentPlayer.baseXPos = input.readDouble(); // sectorX
currentPlayer.baseYPos = input.readDouble(); // sectorY
currentPlayer.alliance = input.readUTF(); // allianceKey
currentPlayer.allianceName = input.readUTF(); // allianceName
currentPlayer.isAllianceOpen = input.readBoolean(); // alliance is public
input.checkToken();
players.push(currentPlayer);
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
```

```
players = null;
}
}
}

------------------------------------
File 518: igw\com\service\server\incoming\leaderboard\WarfrontUpdateResponse.as
package com.service.server.incoming.leaderboard
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.WarfrontData;

import org.shared.ObjectPool;

public class WarfrontUpdateResponse implements IResponse
{
public var removed:Vector.<String> = new Vector.<String>;
public var warfronts:Vector.<WarfrontData> = new Vector.<WarfrontData>;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
var data:WarfrontData;
var num:int = input.readUnsignedInt();
for (var idx:int = 0; idx < num; idx++)
{
input.readUTF(); // map key
data = ObjectPool.get(WarfrontData);
data.read(input);
warfronts.push(data);
}
num = input.readUnsignedInt();
for (idx = 0; idx < num; idx++)
{
removed.push(input.readUTF());
/*reason =*/ input.readInt();
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public
```

```
function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
for (var i:int = 0; i < warfronts.length; i++)
ObjectPool.give(warfronts[i]);
warfronts.length = 0;
}
}
}

-----------------------------------
File 519: igw\com\service\server\incoming\mail\MailDetailResponse.as
package com.service.server.incoming.mail
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class MailDetailResponse implements IResponse
{
public var key:String;
public var sender:String;
public var senderAlliance:String;
public var senderRace:String;
public var body:String;
public var html:Boolean;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
key = input.readUTF();
sender = input.readUTF();
senderAlliance = input.readUTF();
senderRace = input.readUTF();
body = input.readUTF();
html = input.readBoolean();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}
```

```
public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}

-------------------------------------
File 520: igw\com\service\server\incoming\mail\MailInboxResponse.as
package com.service.server.incoming.mail
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.MailInboxData;

import org.shared.ObjectPool;

public class MailInboxResponse implements IResponse
{
public var nowMillis:Number;
public var mailData:Vector.<MailInboxData> = new Vector.<MailInboxData>;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
nowMillis = input.readInt64();

var mailInboxData:MailInboxData;
var numMails:int = input.readUnsignedInt();
for ( var idx:int = 0; idx < numMails; ++idx )
{
// TODO - mail object, this reads summary fields for inbox display
mailInboxData = ObjectPool.get(MailInboxData);
mailInboxData.read(input);
mailData.push(mailInboxData);
}
input.checkToken();
}

public
```

```
function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}

-----------------------------------
File 521: igw\com\service\server\incoming\mail\MailUnreadResponse.as
package com.service.server.incoming.mail
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class MailUnreadResponse implements IResponse
{
public var unread:int;
public var total:int;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
unread = input.readInt();
total = input.readInt();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public
```

```
function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 522: igw\com\service\server\incoming\proxy\AuthResponse.as

```
package com.service.server.incoming.proxy
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class AuthResponse implements IResponse
{
public function AuthResponse()
{
}

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
}

public function readJSON( input:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return 0; }
public function set header( v:int ):void {}

public function get protocolID():int { return 0; }
public function set protocolID( v:int ):void {}

public function destroy():void
{
}
}
}
```

File 523: igw\com\service\server\incoming\proxy\ProxyBattleDisconnectedResponse.as

```
package com.service.server.incoming.proxy
{
import
```

```
com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class ProxyBattleDisconnectedResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
/* disconnected = */
input.readBoolean(); // always true
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}

-----------------------------------
File 524: igw\com\service\server\incoming\proxy\ProxySectorDisconnectedResponse.as
package com.service.server.incoming.proxy
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class ProxySectorDisconnectedResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
/* disconnected = */
input.readBoolean();
```

```
// always true
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}

-------------------------------------
File 525: igw\com\service\server\incoming\proxy\ProxyStarbaseDisconnectedResponse.as
package com.service.server.incoming.proxy
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class ProxyStarbaseDisconnectedResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
/* disconnected = */
input.readBoolean(); // always true
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public
```

```
function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 526: igw\com\service\server\incoming\proxy\TimeSyncResponse.as

```
package com.service.server.incoming.proxy
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class TimeSyncResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
var clientNow:int = input.readInt();
var serverNow:Number = input.readInt64();
/*ServerTime.CLIENT_TIME = getTimer();
ServerTime.LATENCY = (ServerTime.CLIENT_TIME - clientNow) / 2;
ServerTime.SERVER_TIME = ReadUtil.readSignedFixedInt64(input).toNumber() +
ServerTime.LATENCY;*/
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 527: igw\com\service\server\incoming\sector\SectorAlwaysVisibleBaselineResponse.as

```
package com.service.server.incoming.sector
{
import com.model.player.PlayerVO;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;
import com.service.server.incoming.data.IServerData;
import com.service.server.incoming.data.SectorBattleData;
import com.service.server.incoming.data.SectorData;
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorObjectiveData;
import com.service.server.incoming.data.SectorOrderData;

import org.shared.ObjectPool;

public class SectorAlwaysVisibleBaselineResponse implements ITickedResponse
{
public var activeSplitPrototypes:Vector.<String> = new Vector.<String>;

public var players:Vector.<PlayerVO> = new Vector.<PlayerVO>;
public var battles:Vector.<SectorBattleData> = new Vector.<SectorBattleData>;
public var entities:Vector.<SectorEntityData> = new Vector.<SectorEntityData>;
public var orders:Vector.<SectorOrderData> = new Vector.<SectorOrderData>;
public var objectives:Vector.<SectorObjectiveData> = new Vector.<SectorObjectiveData>;
public var sector:SectorData;

private var _header:int;
private var _protocolID:int;
private var _tick:int;
private var _timeStep:int;

public function read( input:BinaryInputStream ):void
{
input.setStringInputCache(input.sectorAlwaysVisibleStringInputCache);
input.readStringCacheBaseline();

input.checkToken();

var activeSplit:String;
var numSplits:int = input.readUnsignedInt();
for (i = 0; i < numSplits; ++i)
{
activeSplit = input.readUTF(); // split test prototype
if (activeSplit != '')
activeSplitPrototypes.push(activeSplit);
}


_tick
```

```
= input.readInt();
_timeStep = input.readInt();

sector = ObjectPool.get(SectorData);
sector.id = input.readUTF();
sector.prototype = PrototypeModel.instance.getSectorPrototypeByName(input.readUTF());
sector.appearanceSeed = input.readInt();
sector.sectorName =
PrototypeModel.instance.getSectorNamePrototypeByName(input.readUTF());
sector.sectorEnum =
PrototypeModel.instance.getSectorNamePrototypeByName(input.readUTF());
sector.neighborhood = input.readInt64();

var data:IServerData;
var vo:PlayerVO;
//get the players
var numPlayers:int = input.readUnsignedInt();
for (var i:int = 0; i < numPlayers; i++)
{
input.readUTF();
vo = ObjectPool.get(PlayerVO);
vo.read(input);
if(vo.name.length > 0)
players.push(vo);
}
var numEntities:int = input.readUnsignedInt();
for (i = 0; i < numEntities; i++)
{
input.readUTF();
data = ObjectPool.get(SectorEntityData);
data.read(input);
entities.push(data);
}
var numBattles:int = input.readUnsignedInt();
for (i = 0; i < numBattles; i++)
{
input.readUTF();
data = ObjectPool.get(SectorBattleData);
data.read(input);
battles.push(data);
}
var numOrders:int = input.readUnsignedInt();
for (i = 0; i < numOrders; i++)
{
input.readUTF();
data = ObjectPool.get(SectorOrderData);
data.read(input);
orders.push(data);
}
//get the waypoints
var
```

```
numWaypoints:int = input.readUnsignedInt();
for (i = 0; i < numWaypoints; i++)
{
//input.readUTF();
data = ObjectPool.get(SectorObjectiveData);
data.read(input);
objectives.push(data);
// var str:String = input.readUTF(); //Mission key
// input.checkToken();
// var x:Number = input.readDouble(); //locationX
// var y:Number = input.readDouble(); //locationY
// input.readUTF(); //Asset
// input.readUTF(); //Type
// input.checkToken();
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in SectorAlwaysVisibleResponse is not supported");
}

public function get isBaseline():Boolean { return false; } // while this is 'a' baseline, it is not first
baseline packet
public function get isTicked():Boolean { return false; }
public function get addTick():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public function destroy():void
{
ObjectPool.give(sector);
sector = null;

var i:int = 0;
players.length = 0;
for (i = 0; i < entities.length; i++)
ObjectPool.give(entities[i]);
entities.length = 0;
for (i = 0; i < battles.length; i++)
ObjectPool.give(battles[i]);
battles.length = 0;
for
```

```
(i = 0; i < orders.length; i++)
ObjectPool.give(orders[i]);
orders.length = 0;
}
}
}

------------------------------------
File 528: igw\com\service\server\incoming\sector\SectorAlwaysVisibleUpdateResponse.as
package com.service.server.incoming.sector
{
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;
import com.service.server.incoming.data.IServerData;
import com.service.server.incoming.data.RemovedObjectData;
import com.service.server.incoming.data.SectorBattleData;
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorEntityUpdateData;
import com.service.server.incoming.data.SectorObjectiveData;
import com.service.server.incoming.data.SectorOrderData;

import org.shared.ObjectPool;

public class SectorAlwaysVisibleUpdateResponse implements ITickedResponse
{
public var players:Vector.<PlayerVO> = new Vector.<PlayerVO>;
public var entities:Vector.<SectorEntityData> = new Vector.<SectorEntityData>;
public var battles:Vector.<SectorBattleData> = new Vector.<SectorBattleData>;
public var orders:Vector.<SectorOrderData> = new Vector.<SectorOrderData>;
public var objectives:Vector.<SectorObjectiveData> = new Vector.<SectorObjectiveData>;

public var removedPlayers:Vector.<RemovedObjectData> = new
Vector.<RemovedObjectData>;
public var removedEntities:Vector.<RemovedObjectData> = new
Vector.<RemovedObjectData>;
public var removedBattles:Vector.<RemovedObjectData> = new Vector.<RemovedObjectData>;
public var removedOrders:Vector.<RemovedObjectData> = new Vector.<RemovedObjectData>;
public var removedObjectives:Vector.<RemovedObjectData> = new
Vector.<RemovedObjectData>;

public var entityUpdates:Vector.<SectorEntityUpdateData> = new
Vector.<SectorEntityUpdateData>;

private var _header:int;
private var _protocolID:int;
private var _tick:int;
private var _timeStep:int;

public
```

```
function read( input:BinaryInputStream ):void
{
input.setStringInputCache(input.sectorAlwaysVisibleStringInputCache);

input.checkToken();
_tick = input.readInt();
_timeStep = input.readInt();

var data:IServerData;
var vo:PlayerVO;
//get the players
var numPlayers:int = input.readUnsignedInt();
for (var i:int = 0; i < numPlayers; i++)
{
input.readUTF();
vo = ObjectPool.get(PlayerVO);
vo.read(input);
players.push(vo);
}
//get the entities
var numEntities:int = input.readUnsignedInt();
for (i = 0; i < numEntities; i++)
{
input.readUTF();
data = ObjectPool.get(SectorEntityData);
data.read(input);
entities.push(data);
}
//get the battles
var numBattles:int = input.readUnsignedInt();
for (i = 0; i < numBattles; i++)
{
input.readUTF();
data = ObjectPool.get(SectorBattleData);
data.read(input);
battles.push(data);
}
//get the orders
var numOrders:int = input.readUnsignedInt();
for (i = 0; i < numOrders; i++)
{
input.readUTF();
data = ObjectPool.get(SectorOrderData);
data.read(input);
orders.push(data);
}
//get the waypoints
var numWaypoints:int = input.readUnsignedInt();
for (i = 0; i < numWaypoints; i++)
{
//input.readUTF();
```

```
data = ObjectPool.get(SectorObjectiveData);
data.read(input);
objectives.push(data);
// input.readUTF(); //Mission key
// input.checkToken();
// input.readDouble(); //locationX
// input.readDouble(); //locationY
// input.readUTF(); //waypoint
// input.readUTF(); //Type
// input.checkToken();
}

//get the removed players
numPlayers = input.readUnsignedInt();
for (i = 0; i < numPlayers; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedPlayers.push(data);
}
//get the removed entities
numEntities = input.readUnsignedInt();
for (i = 0; i < numEntities; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedEntities.push(data);
}
//get the removed battles
numBattles = input.readUnsignedInt();
for (i = 0; i < numBattles; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedBattles.push(data);
}
//get the removed orders
numOrders = input.readUnsignedInt();
for (i = 0; i < numOrders; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedOrders.push(data);
}
//get the removed waypoints
numWaypoints = input.readUnsignedInt();
for (i = 0; i < numWaypoints; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
```

```
removedObjectives.push(data);
}

var numUpdates:int = input.readUnsignedInt();
for (i = 0; i < numUpdates; i++)
{
data = ObjectPool.get(SectorEntityUpdateData);
SectorEntityUpdateData(data).id = input.readUTF();
SectorEntityUpdateData(data).read(input);
entityUpdates.push(data);
}

numUpdates = input.readUnsignedInt();
for (i = 0; i < numUpdates; i++)
{
var playerKey:String = input.readUTF();
input.checkToken();
var xp:uint = input.readInt64();
var name:String = input.readUTF();
var alliance:String = input.readUTF();
var allianceName:String = input.readUTF();
input.checkToken();
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("this is no longer supported");
}

public function get isBaseline():Boolean { return false; }
public function get isTicked():Boolean { return true; }
public function get addTick():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public function destroy():void
{
var i:int = 0;
players.length = 0;
for (i = 0; i < entities.length; i++)
ObjectPool.give(entities[i]);
```

```
entities.length = 0;
for (i = 0; i < battles.length; i++)
ObjectPool.give(battles[i]);
battles.length = 0;
for (i = 0; i < orders.length; i++)
ObjectPool.give(orders[i]);
orders.length = 0;

for (i = 0; i < removedPlayers.length; i++)
ObjectPool.give(removedPlayers[i]);
removedPlayers.length = 0;
for (i = 0; i < removedEntities.length; i++)
ObjectPool.give(removedEntities[i]);
removedEntities.length = 0;
for (i = 0; i < removedBattles.length; i++)
ObjectPool.give(removedBattles[i]);
removedBattles.length = 0;
for (i = 0; i < removedOrders.length; i++)
ObjectPool.give(removedOrders[i]);
removedOrders.length = 0;
for (i = 0; i < removedObjectives.length; i++)
ObjectPool.give(removedObjectives[i]);
removedObjectives.length = 0;

for (i = 0; i < entityUpdates.length; i++)
ObjectPool.give(entityUpdates[i]);
entityUpdates.length = 0;
}
}
}

------------------------------------
File 529: igw\com\service\server\incoming\sector\SectorBaselineResponse.as
package com.service.server.incoming.sector
{
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;
import com.service.server.incoming.data.IServerData;
import com.service.server.incoming.data.SectorBattleData;
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorOrderData;

import org.shared.ObjectPool;

public class SectorBaselineResponse implements ITickedResponse
{
public var battles:Vector.<SectorBattleData> = new Vector.<SectorBattleData>;
public var entities:Vector.<SectorEntityData> = new Vector.<SectorEntityData>;
public var orders:Vector.<SectorOrderData> = new Vector.<SectorOrderData>;

private
```

```
var _header:int;
private var _protocolID:int;
private var _tick:int;
private var _timeStep:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
_tick = input.readInt();
_timeStep = input.readInt();

var data:IServerData;
var numEntities:int = input.readUnsignedInt();
for (var i:int = 0; i < numEntities; i++)
{
input.readUTF();
data = ObjectPool.get(SectorEntityData);
data.read(input);
entities.push(data);
}
var numBattles:int = input.readUnsignedInt();
for (i = 0; i < numBattles; i++)
{
input.readUTF();
data = ObjectPool.get(SectorBattleData);
data.read(input);
battles.push(data);
}
var numOrders:int = input.readUnsignedInt();
for (i = 0; i < numOrders; i++)
{
input.readUTF();
data = ObjectPool.get(SectorOrderData);
data.read(input);
orders.push(data);
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
_tick = data.nowTick;
_timeStep = data.tickTimeMillis;

var obj:IServerData;
var key:String
for (key in data.Entities)
{
obj = ObjectPool.get(SectorEntityData);
obj.readJSON(data.Entities[key]);
entities.push(obj);
```

```
}
for (key in data.Battles)
{
obj = ObjectPool.get(SectorBattleData);
obj.readJSON(data.Battles[key]);
battles.push(obj);
}
for (key in data.Orders)
{
obj = ObjectPool.get(SectorOrderData);
obj.readJSON(data.Orders[key]);
orders.push(obj);
}
}

public function get isBaseline():Boolean { return true; }
public function get isTicked():Boolean { return true; }
public function get addTick():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public function destroy():void
{
var i:int = 0;
for (i = 0; i < entities.length; i++)
ObjectPool.give(entities[i]);
entities.length = 0;
for (i = 0; i < battles.length; i++)
ObjectPool.give(battles[i]);
battles.length = 0;
for (i = 0; i < orders.length; i++)
ObjectPool.give(orders[i]);
orders.length = 0;
}
}
}

-----------------------------------
File 530: igw\com\service\server\incoming\sector\SectorFleetTravelAlertResponse.as
package com.service.server.incoming.sector
{
import com.service.server.BinaryInputStream;
import
```

```
com.service.server.IResponse;

public class SectorFleetTravelAlertResponse implements IResponse
{
public var entityId:String;
public var sectorKey:String;
private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
}

public function readJSON( data:Object ):void
{
entityId = data.entityId;
sectorKey = data.sectorKey;
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}

-----------------------------------
File 531: igw\com\service\server\incoming\sector\SectorUpdateResponse.as
package com.service.server.incoming.sector
{
import com.service.server.BinaryInputStream;
import com.service.server.ITickedResponse;
import com.service.server.incoming.data.IServerData;
import com.service.server.incoming.data.RemovedObjectData;
import com.service.server.incoming.data.SectorBattleData;
import com.service.server.incoming.data.SectorEntityData;
import com.service.server.incoming.data.SectorEntityUpdateData;
import com.service.server.incoming.data.SectorOrderData;

import org.shared.ObjectPool;

public
```

```
class SectorUpdateResponse implements ITickedResponse
{
public var entities:Vector.<SectorEntityData> = new Vector.<SectorEntityData>;
public var battles:Vector.<SectorBattleData> = new Vector.<SectorBattleData>;
public var orders:Vector.<SectorOrderData> = new Vector.<SectorOrderData>;

public var removedEntities:Vector.<RemovedObjectData> = new
Vector.<RemovedObjectData>;
public var removedBattles:Vector.<RemovedObjectData> = new Vector.<RemovedObjectData>;
public var removedOrders:Vector.<RemovedObjectData> = new Vector.<RemovedObjectData>;

public var entityUpdates:Vector.<SectorEntityUpdateData> = new
Vector.<SectorEntityUpdateData>;

private var _header:int;
private var _protocolID:int;
private var _tick:int;
private var _timeStep:int;

public function read( input:BinaryInputStream ):void
{
_tick = input.readInt();
_timeStep = input.readInt();

var numQuadrants:int = input.readByte();
for (var q:int = 0; q < numQuadrants; q++)
{
input.checkToken();
var data:IServerData;
//get the entities
var numEntities:int = input.readUnsignedInt();
for (var i:int = 0; i < numEntities; i++)
{
data = ObjectPool.get(SectorEntityData);
data.read(input);
entities.push(data);
}
//get the battles
var numBattles:int = input.readUnsignedInt();
for (i = 0; i < numBattles; i++)
{
data = ObjectPool.get(SectorBattleData);
data.read(input);
battles.push(data);
}
//get the orders
var numOrders:int = input.readUnsignedInt();
for (i = 0; i < numOrders; i++)
{
data = ObjectPool.get(SectorOrderData);
data.read(input);
```

```
orders.push(data);
}

//get the removed entities
numEntities = input.readUnsignedInt();
for (i = 0; i < numEntities; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedEntities.push(data);
}
//get the removed battles
numBattles = input.readUnsignedInt();
for (i = 0; i < numBattles; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedBattles.push(data);
}
//get the removed orders
numOrders = input.readUnsignedInt();
for (i = 0; i < numOrders; i++)
{
data = ObjectPool.get(RemovedObjectData);
RemovedObjectData(data).read(input);
removedOrders.push(data);
}

var numUpdates:int = input.readUnsignedInt();
for (i = 0; i < numUpdates; i++)
{
data = ObjectPool.get(SectorEntityUpdateData);
SectorEntityUpdateData(data).id = input.readUTF();
SectorEntityUpdateData(data).read(input);
entityUpdates.push(data);
}
input.checkToken();
}
}

public function readJSON( data:Object ):void
{
throw new Error("this is unsupported");
}

public function get isBaseline():Boolean { return false; }
public function get isTicked():Boolean { return true; }
public function get addTick():Boolean { return true; }

public
```

```
function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function get tick():int { return _tick }
public function get timeStep():int { return _timeStep; }

public function destroy():void
{
var i:int = 0;
for (i = 0; i < entities.length; i++)
ObjectPool.give(entities[i]);
entities.length = 0;
for (i = 0; i < battles.length; i++)
ObjectPool.give(battles[i]);
battles.length = 0;
for (i = 0; i < orders.length; i++)
ObjectPool.give(orders[i]);
orders.length = 0;

for (i = 0; i < removedEntities.length; i++)
ObjectPool.give(removedEntities[i]);
removedEntities.length = 0;
for (i = 0; i < removedBattles.length; i++)
ObjectPool.give(removedBattles[i]);
removedBattles.length = 0;
for (i = 0; i < removedOrders.length; i++)
ObjectPool.give(removedOrders[i]);
removedOrders.length = 0;

for (i = 0; i < entityUpdates.length; i++)
ObjectPool.give(entityUpdates[i]);
entityUpdates.length = 0;
}
}
}

------------------------------------
File 532: igw\com\service\server\incoming\starbase\StarbaseAchievementsResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.AchievementData;
import com.service.server.incoming.data.ScoreData;

import org.shared.ObjectPool;

public
```

```
class StarbaseAchievementsResponse implements IResponse
{
public var unlockToast:Boolean;
public var achievements:Vector.<AchievementData> = new Vector.<AchievementData>;
public var scores:Vector.<ScoreData> = new Vector.<ScoreData>;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

unlockToast = input.readBoolean();

var i:uint
var achievementData:AchievementData;
var len:uint = input.readUnsignedInt();
for (i = 0; i < len; ++i)
{
achievementData = ObjectPool.get(AchievementData);
input.readUTF(); // key
achievementData.read(input);
achievements.push(achievementData);
}

var scoreData:ScoreData;
len = input.readUnsignedInt();
for (i = 0; i < len; ++i)
{
scoreData = ObjectPool.get(ScoreData);
input.readUTF(); // key
scoreData.read(input);
scores.push(scoreData);
}

input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public
```

```
function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}
```

File 533: igw\com\service\server\incoming\starbase\StarbaseAllScoresResponse.as

```
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.ScoreData;
import com.service.server.incoming.data.MissionScoreData;

import org.shared.ObjectPool;

public class StarbaseAllScoresResponse implements IResponse
{
public var scores:Vector.<ScoreData> = new Vector.<ScoreData>;
public var missionScores:Vector.<MissionScoreData> = new Vector.<MissionScoreData>;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

var i:uint;
var scoreData:ScoreData;
var len:uint = input.readUnsignedInt();
for (i = 0; i < len; ++i)
{
scoreData = ObjectPool.get(ScoreData);
input.readUTF(); // key
scoreData.read(input);
scores.push(scoreData);
}

var missionScoreData:MissionScoreData;
var len:uint = input.readUnsignedInt();
for (i = 0; i < len; ++i)
{
missionScoreData = ObjectPool.get(MissionScoreData);
input.readUTF(); // key
missionScoreData.read(input);
missionScores.push(missionScoreData);
```

```
}

input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 534: igw\com\service\server\incoming\starbase\StarbaseAvailableRerollResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseAvailableRerollResponse implements IResponse
{
public var battleKey:String;
public var blueprintPrototype:String;
public var crewMemberPrototype:String;
public var timeoutDelta:Number;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
battleKey = input.readUTF(); // which battle this reroll is for
blueprintPrototype = input.readUTF(); // set if you won a blueprint, that means this reroll allows
you
```

```
to
timeoutDelta = input.readInt64();
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 535: igw\com\service\server\incoming\starbase\StarbaseBaselineResponse.as
package com.service.server.incoming.starbase
{
import com.enum.CurrencyEnum;
import com.model.player.CurrentUser;
import com.model.player.OfferVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.BaseData;
import com.service.server.incoming.data.BlueprintData;
import com.service.server.incoming.data.BookmarksData;
import com.service.server.incoming.data.BuffData;
import com.service.server.incoming.data.BuildingData;
import com.service.server.incoming.data.CrewMemberData;
import com.service.server.incoming.data.FleetData;
import com.service.server.incoming.data.MissionData;
import com.service.server.incoming.data.ResearchData;
import com.service.server.incoming.data.ShipData;
import com.service.server.incoming.data.TradeRouteData;

import flash.utils.getTimer;

import
```

```
org.shared.ObjectPool;

public class StarbaseBaselineResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var validData:Boolean = false;

public var bases:Vector.<BaseData> = new Vector.<BaseData>;
public var blueprints:Vector.<BlueprintData> = new Vector.<BlueprintData>;
public var buffs:Vector.<BuffData> = new Vector.<BuffData>;
public var buildings:Vector.<BuildingData> = new Vector.<BuildingData>;
public var fleets:Vector.<FleetData> = new Vector.<FleetData>;
public var missions:Vector.<MissionData> = new Vector.<MissionData>;
public var research:Vector.<ResearchData> = new Vector.<ResearchData>;
public var ships:Vector.<ShipData> = new Vector.<ShipData>;
public var tradeRoutes:Vector.<TradeRouteData> = new Vector.<TradeRouteData>;
public var bookmarks:BookmarksData = new BookmarksData();
public var upcomingEvents:Vector.<IPrototype> = new Vector.<IPrototype>;
public var activeEvents:Vector.<IPrototype> = new Vector.<IPrototype>;
public var activeSplitPrototypes:Vector.<String> = new Vector.<String>;
public var settings:Object;
public var update:Boolean;
public var updateReason:String;
public var nowMillis:Number;
public var baselineType:Number;

public static const BASELINE_ALL:Number = 0;
public static const BASELINE_PLAYER:Number = 1 << 0;
public static const BASELINE_SHIP:Number = 1 << 1;
public static const BASELINE_FLEET:Number = 1 << 2;
public static const BASELINE_MISSION:Number = 1 << 3;
public static const BASELINE_BLUEPRINT:Number = 1 << 4;
public static const BASELINE_CREW:Number = 1 << 5;
public static const BASELINE_BASE:Number = 1 << 6;
public static const BASELINE_BUILDING:Number = 1 << 7;
public static const BASELINE_RESEARCH:Number = 1 << 8;
public static const BASELINE_TRADE_ROUTE:Number = 1 << 9;
public static const BASELINE_BUFF:Number = 1 << 10;
public static const BASELINE_OFFER:Number = 1 << 11;
public static const BASELINE_SETTING:Number = 1 << 12;

public function read( input:BinaryInputStream ):void
{
var prototypeModel:PrototypeModel = PrototypeModel.instance;

input.setStringInputCache(input.starbaseBaselineStringInputCache);
input.checkToken();

if(!input.validToken)
```

```
return;

update = input.readBoolean();
var i:int = 0;
if (!update)
{
input.readStringCacheBaseline();

var activeSplit:String;
var numSplits:int = input.readUnsignedInt();
for (i = 0; i < numSplits; ++i)
{
activeSplit = input.readUTF(); // split test prototype
if (activeSplit != '')
activeSplitPrototypes.push(activeSplit);
}
}

nowMillis = input.readInt64();

if (!update)
{
// get event prototypes
var eventProtos:Vector.<IPrototype> = prototypeModel.getEventPrototypes();
var event:IPrototype;

var begins:Number;
var ends:Number;
for (i = 0; i < eventProtos.length; ++i)
{
// now... stuff
event = eventProtos[i];

begins = event.getValue("eventBegins");
ends = event.getValue("eventEnds");
if (ends < nowMillis)
continue;
if (begins > nowMillis)
{
upcomingEvents.push(event);
// this event starts in the future
// timer ? start at (begins - nowMillis)
// TODO
} else
{
activeEvents.push(event);
// this event is happening now
// timer to end event at (ends - nowMillis)
// apply buffs to buff calcs
//
```

```
TODO
}
}
}

updateReason = input.readUTF();

baselineType = input.readInt64();

if(!input.validToken)
return;

// players
if(baselineType == BASELINE_ALL || baselineType & BASELINE_PLAYER)
{
// there should only be one entry in this map
var numPlayers:int = input.readUnsignedInt();
for (i = 0; i < numPlayers; ++i)
{
input.readUTF(); // key
input.checkToken();
input.checkToken();
input.readUTF(); // player key (already assigned to CurrentUser.id from login)
input.checkToken();

CurrentUser.faction = input.readUTF();
CurrentUser.avatarName = input.readUTF(); // (racePrototype)
CurrentUser.name = input.readUTF();
CurrentUser.xp = input.readInt64();
var premium:uint = input.readInt64();
if (update && CurrentUser.wallet.premium < premium)
CurrentUser.wallet.deposit(premium - CurrentUser.wallet.premium, CurrencyEnum.PREMIUM);
else
CurrentUser.wallet.overridePremium = premium;

CurrentUser.alliance = input.readUTF();

CurrentUser.homeBase = input.readUTF();
CurrentUser.centerSpaceBase = input.readUTF();
bookmarks.read(input);

CurrentUser.commendationPointsPVE = input.readInt64();
CurrentUser.commendationPointsPVP = input.readInt64();

CurrentUser.allianceName = input.readUTF(); // allianceName

CurrentUser.purchasedShipSlots = input.readInt64();

input.checkToken();
}

CurrentUser.playerWalletKey
```

```
= input.readUTF();

if(!input.validToken)
return;
}

// ships

if(baselineType == BASELINE_ALL || baselineType & BASELINE_SHIP)
{
var shipData:ShipData;
var numShips:int = input.readUnsignedInt();
for (i = 0; i < numShips; i++)
{
shipData = ObjectPool.get(ShipData);
input.readUTF(); // key
shipData.read(input);
ships.push(shipData);
}

if(!input.validToken)
return;
}

// fleets

if(baselineType == BASELINE_ALL || baselineType & BASELINE_FLEET)
{
var fleetData:FleetData;
var numFleets:int = input.readUnsignedInt();
for (i = 0; i < numFleets; i++)
{
fleetData = ObjectPool.get(FleetData);
input.readUTF(); // key
fleetData.read(input);
fleets.push(fleetData);
}

fleets.sort(fleet_comparator);

if(!input.validToken)
return;
}

// missions

if(baselineType == BASELINE_ALL || baselineType & BASELINE_MISSION)
{
var missionData:MissionData;
var numMissions:int = input.readUnsignedInt();
for
```

```
(i = 0; i < numMissions; ++i)
{
missionData = ObjectPool.get(MissionData);
input.readUTF(); // key
missionData.read(input);
missionData.prototype = prototypeModel.getMissionPrototye(missionData.missionPrototype);
missions.push(missionData);
}

if(!input.validToken)
return;
}

// blueprints
if(baselineType == BASELINE_ALL || baselineType & BASELINE_BLUEPRINT)
{
var blueprintData:BlueprintData;
var numBlueprints:int = input.readUnsignedInt();
for (i = 0; i < numBlueprints; ++i)
{
blueprintData = ObjectPool.get(BlueprintData);
input.readUTF(); // key
blueprintData.read(input);
blueprintData.prototype =
prototypeModel.getBlueprintPrototype(blueprintData.blueprintPrototype);
blueprints.push(blueprintData);
}

if(!input.validToken)
return;
}

if(baselineType == BASELINE_ALL || baselineType & BASELINE_CREW)
{
var crewMemberData:CrewMemberData;
var numCrewMembers:int = input.readUnsignedInt();
for (i = 0; i < numCrewMembers; ++i)
{
crewMemberData = ObjectPool.get(CrewMemberData);
input.readUTF(); // key
crewMemberData.read(input);
//Do whatever else you need to do here, I guess.
}

if(!input.validToken)
return;
}

// bases
if(baselineType == BASELINE_ALL || baselineType & BASELINE_BASE)
{
```

```
var baseData:BaseData;
var numBases:int = input.readUnsignedInt();
var timeRemaining:Number;
for (i = 0; i < numBases; ++i)
{
baseData = ObjectPool.get(BaseData);
input.readUTF(); // key
baseData.read(input);
baseData.lastUpdateTimeUTCMillis = getTimer() - (nowMillis -
baseData.lastUpdateTimeUTCMillis);
timeRemaining = baseData.bubbleEnds - nowMillis;
baseData.bubbleTimeRemaining = (timeRemaining > 0) ? timeRemaining : 0;
bases.push(baseData);
}

if(!input.validToken)
return;
}

// buildings
if(baselineType == BASELINE_ALL || baselineType & BASELINE_BUILDING)
{
var buildingData:BuildingData;
var numBuildings:int = input.readUnsignedInt();
for (i = 0; i < numBuildings; ++i)
{
buildingData = ObjectPool.get(BuildingData);
input.readUTF(); // key
buildingData.read(input);
if (buildingData.prototype)
buildings.push(buildingData);
}

if(!input.validToken)
return;
}

// research
if(baselineType == BASELINE_ALL || baselineType & BASELINE_RESEARCH)
{
var researchData:ResearchData;
var numResearch:int = input.readUnsignedInt();
for (i = 0; i < numResearch; ++i)
{
researchData = ObjectPool.get(ResearchData);
input.readUTF(); // key
researchData.read(input);
research.push(researchData);
}

if(!input.validToken)
```

```
return;
}

// trade routes
if(baselineType == BASELINE_ALL || baselineType & BASELINE_TRADE_ROUTE)
{
var tradeRouteData:TradeRouteData;
var numTradeRoutes:int = input.readUnsignedInt();
for (i = 0; i < numTradeRoutes; ++i)
{
tradeRouteData = ObjectPool.get(TradeRouteData);
input.readUTF(); // key
tradeRouteData.read(input);
tradeRoutes.push(tradeRouteData);
}

if(!input.validToken)
return;
}

// buffs
if(baselineType == BASELINE_ALL || baselineType & BASELINE_BUFF)
{
var buffData:BuffData;
var numBuffs:int = input.readUnsignedInt();
for (i = 0; i < numBuffs; ++i)
{
buffData = ObjectPool.get(BuffData);
input.readUTF(); // key
buffData.read(input);
buffData.now = nowMillis;
buffs.push(buffData);
}

if(!input.validToken)
return;
}

// offers
if(baselineType == BASELINE_ALL || baselineType & BASELINE_OFFER)
{
var offers:Vector.<OfferVO> = new Vector.<OfferVO>;
var currentOffer:OfferVO;
var numOffers:int = input.readUnsignedInt();
for (i = 0; i < numOffers; ++i)
{
currentOffer = new OfferVO(input.readUTF(), input.readInt64(), input.readInt64());
offers.push(currentOffer)
}

if(!input.validToken)
```

```
return;

CurrentUser.offers = offers;
}

// settings
if(baselineType == BASELINE_ALL || baselineType & BASELINE_SETTING)
{
var numSettings:int = input.readUnsignedInt();
for (i = 0; i < numSettings; ++i)
{
input.readUTF(); // key
input.checkToken();
input.checkToken();
/*id =*/
input.readUTF();
input.checkToken();
/* playerOwnerID = */
input.readUTF();
var settingsText:String = input.readUTF();
input.checkToken();

try
{
if (settingsText != '')
settings = JSON.parse(settingsText);
} catch ( e:Error )
{

}

// as with player data, we only expect one entry in this map
}

if(!input.validToken)
return;
}

input.checkToken();

if(!input.validToken)
return;

validData = true;
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON in StarbaseBaselineResponse is not supported");
}
```

```
public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
//cleanup bases
for (var i:int = 0; i < bases.length; i++)
ObjectPool.give(bases[i]);
bases.length = 0;
//cleanup blueprints
for (i = 0; i < blueprints.length; i++)
ObjectPool.give(blueprints[i]);
blueprints.length = 0;
//cleanup buffs
for (i = 0; i < buffs.length; i++)
ObjectPool.give(buffs[i]);
buffs.length = 0;
//cleanup buildings
for (i = 0; i < buildings.length; i++)
ObjectPool.give(buildings[i]);
buildings.length = 0;
//cleanup fleets
for (i = 0; i < fleets.length; i++)
ObjectPool.give(fleets[i]);
fleets.length = 0;
//cleanup missions
for (i = 0; i < missions.length; i++)
ObjectPool.give(missions[i]);
missions.length = 0;
//cleanup research
for (i = 0; i < research.length; i++)
ObjectPool.give(research[i]);
research.length = 0;
//cleanup ships
for (i = 0; i < ships.length; i++)
ObjectPool.give(ships[i]);
ships.length = 0;
//cleanup traderoutes
for (i = 0; i < tradeRoutes.length; i++)
ObjectPool.give(tradeRoutes[i]);
tradeRoutes.length = 0;
}

private
```

```
function fleet_comparator( f1:FleetData, f2:FleetData ):Number
{
if (f1.id == f2.id)
return 0;
else if (f1.id < f2.id)
return -1;
return 1;
}
}
}
```

File 536: igw\com\service\server\incoming\starbase\StarbaseBattleAlertResponse.as

```
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class StarbaseBattleAlertResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var homeBaseBattle:String; // note that these are all serverIdentifier strings that can be used to ConnectToBattle
public var centerSpaceBaseBattle:String;
public var fleetBattles:Object = new Object;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
homeBaseBattle = input.readUTF();
centerSpaceBaseBattle = input.readUTF();
var mapSize:int = input.readUnsignedInt();
for (var i:int = 0; i < mapSize; i++)
{
fleetBattles[input.readUTF()] = input.readUTF();
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
homeBaseBattle = data.homeBaseBattle;
centerSpaceBaseBattle = data.centerSpaceBaseBattle;
fleetBattles = data.Battles;
}

public
```

```
function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
fleetBattles = null;
}
}
}

-------------------------------------
File 537: igw\com\service\server\incoming\starbase\StarbaseBountyRewardResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class StarbaseBountyRewardResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var bounty:Number;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
bounty = input.readInt64();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public
```

```
function destroy():void
{
}
}
}

------------------------------------
File 538: igw\com\service\server\incoming\starbase\StarbaseDailyResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseDailyResponse implements IResponse
{
public var escalation:int;
public var canNextClaimDelta:Number;
public var dailyResetsDelta:Number;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
escalation = input.readInt(); // level of daily reward available
canNextClaimDelta = input.readInt64(); // if this is > 0, it's number of milliseconds until you can
claim again
dailyResetsDelta = input.readInt64(); // if this is > 0, your daily escalation will be reset to 0 after
this number of milliseconds
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
```

```
}
}
}

------------------------------------
File 539: igw\com\service\server\incoming\starbase\StarbaseDailyRewardResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseDailyRewardResponse implements IResponse
{
public var alloyReward:Number;
public var creditsReward:Number;
public var energyReward:Number;
public var syntheticReward:Number;
public var blueprintPrototype:String;
public var buffPrototype:String;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
alloyReward = input.readInt64();
creditsReward = input.readInt64();
energyReward = input.readInt64();
syntheticReward = input.readInt64();
buffPrototype = input.readUTF();
blueprintPrototype = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public
```

```
function destroy():void
{
}
}
}

------------------------------------
File 540: igw\com\service\server\incoming\starbase\StarbaseFleetDockedResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class StarbaseFleetDockedResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var fleetPersistence:String;
public var alloyCargo:Number;
public var energyCargo:Number;
public var syntheticCargo:Number;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
fleetPersistence = input.readUTF();
alloyCargo = input.readInt64();
energyCargo = input.readInt64();
syntheticCargo = input.readInt64();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}
```

```
-----------------------------------
File 541: igw\com\service\server\incoming\starbase\StarbaseGetPaywallPayoutsResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseGetPaywallPayoutsResponse implements IResponse
{
public var data:String;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
data = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 542: igw\com\service\server\incoming\starbase\StarbaseInstancedMissionAlertResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import
```

```
com.service.server.IResponse;

public class StarbaseInstancedMissionAlertResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var instancedMissionBattle:String; // note that these are all serverIdentifier strings that can be used to ConnectToBattle

public function read( input:BinaryInputStream ):void
{
input.checkToken();
instancedMissionBattle = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
instancedMissionBattle = data.quickMissionBattle;
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 543: igw\com\service\server\incoming\starbase\StarbaseMissionCompleteResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class StarbaseMissionCompleteResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var missionPersistence:String;
public
```

```
var missionPrototype:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
missionPersistence = input.readUTF();
missionPrototype = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-------------------------------------
File 544: igw\com\service\server\incoming\starbase\StarbaseMotdListResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.MotdData;
import org.shared.ObjectPool;

public class StarbaseMotdListResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var motds:Vector.<MotdData> = new Vector.<MotdData>;

public function read( input:BinaryInputStream ):void
{
input.checkToken();

var motdData:MotdData;
var
```

```
numMotds:int = input.readUnsignedShort();
for ( var idx:int = 0; idx < numMotds; ++idx )
{
motdData = ObjectPool.get(MotdData);
motdData.read(input);
motds.push(motdData);
}

input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 545: igw\com\service\server\incoming\starbase\StarbaseMoveStarbaseResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseMoveStarbaseResponse implements IResponse
{
public var status:int;
/*
enum class MoveBaseStatus
{
Started = 0,
TargetSectorFull = 1,
};
*/

private
```

```
var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
status = input.readInt(); // MoveBaseStatus
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 546: igw\com\service\server\incoming\starbase\StarbaseOfferRedeemedResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

public class StarbaseOfferRedeemedResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public var offerPrototype:String;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
offerPrototype = input.readUTF();
input.checkToken();
}

public function readJSON( data:Object ):void
{
```

```
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}
```

File 547: igw\com\service\server\incoming\starbase\StarbaseRerollChanceResultResponse.as

```
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.CrewMemberData;

import org.shared.ObjectPool;

public class StarbaseRerollChanceResultResponse implements IResponse
{
public var battleKey:String;
public var blueprintPrototype:String;
public var crewMember:CrewMemberData;
public var alloyReward:Number;
public var creditsReward:Number;
public var energyReward:Number;
public var syntheticReward:Number;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
battleKey = input.readUTF(); // which battle this reroll was for
blueprintPrototype = input.readUTF(); // set if you won the reroll, empty if you did not win the
reroll
var hasCrew:Boolean = input.readBoolean();
if (hasCrew)
{
crewMember
```

```
= ObjectPool.get(CrewMemberData);
crewMember.read(input);
}
alloyReward = input.readInt64(); // these are only set if you lost the reroll and win the
'consolation prize'
creditsReward = input.readInt64();
energyReward = input.readInt64();
syntheticReward = input.readInt64();
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File 548: igw\com\service\server\incoming\starbase\StarbaseRerollReceivedResultResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseRerollReceivedResultResponse implements IResponse
{
public var battleKey:String;
public var blueprintPrototype:String;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
```

```
battleKey = input.readUTF(); // which battle this reroll was for
blueprintPrototype = input.readUTF(); // set if you won the reroll, empty if you did not win the reroll
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}
```

File 549: igw\com\service\server\incoming\starbase\StarbaseTransactionResponse.as

```
package com.service.server.incoming.starbase
{
import com.model.transaction.TransactionVO;
import com.service.server.BinaryInputStream;
import com.service.server.incoming.TransactionResponse;

public class StarbaseTransactionResponse extends TransactionResponse
{
override public function read( input:BinaryInputStream ):void
{
super.read(input);
}

override public function readJSON( data:Object ):void
{
super.readJSON(data);
}

override public function destroy():void
{
super.destroy();
}
```

```
}
}

-----------------------------------
File 550: igw\com\service\server\incoming\starbase\StarbaseUnavailableRerollResponse.as
package com.service.server.incoming.starbase
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import org.shared.ObjectPool;

public class StarbaseUnavailableRerollResponse implements IResponse
{
public var battleKey:String;
public var reason:int;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
battleKey = input.readUTF(); // which battle this reroll is for
reason = input.readInt();
input.checkToken();
}

public function readJSON( data:Object ):void
{
// Unimplemented.
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }


public function destroy():void
{
}
}
}

-----------------------------------
File
```

551: igw\com\service\server\incoming\universe\UniverseNeedCharacterCreateResponse.as
```
package com.service.server.incoming.universe
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;

import flash.utils.ByteArray;
import flash.utils.Dictionary;

public class UniverseNeedCharacterCreateResponse implements IResponse
{
private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
input.checkToken();
}

public function readJSON( data:Object ):void
{
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
}
}
}
```

File 552: igw\com\service\server\incoming\universe\UniverseSectorListResponse.as
```
package com.service.server.incoming.universe
{
import com.service.server.BinaryInputStream;
import com.service.server.IResponse;
import com.service.server.incoming.data.SectorData;

import org.shared.ObjectPool;

public class UniverseSectorListResponse implements IResponse
{
public
```

```
var sectors:Vector.<SectorData> = new Vector.<SectorData>;
public var privateSectors:Vector.<SectorData> = new Vector.<SectorData>;

private var _header:int;
private var _protocolID:int;

public function read( input:BinaryInputStream ):void
{
input.checkToken();
{
var numSectors:int = input.readUnsignedInt();
var sectorData:SectorData;
for (var idx:int = 0; idx < numSectors; idx++)
{
/* key */
input.readUTF();

sectorData = ObjectPool.get(SectorData);
sectorData.read(input);
sectors.push(sectorData);
}
}
input.checkToken();
input.checkToken();
{
var numSectors:int = input.readUnsignedInt();
var sectorData:SectorData;
for (var idx:int = 0; idx < numSectors; idx++)
{
/* key */
input.readUTF();

sectorData = ObjectPool.get(SectorData);
sectorData.read(input);
privateSectors.push(sectorData);
}
}
input.checkToken();
}

public function readJSON( data:Object ):void
{
throw new Error("readJSON is not supported");
}

public function get isTicked():Boolean { return false; }

public function get header():int { return _header; }
public function set header( v:int ):void { _header = v; }

public
```

```
function get protocolID():int { return _protocolID; }
public function set protocolID( v:int ):void { _protocolID = v; }

public function destroy():void
{
for (var i:int = 0; i < sectors.length; i++)
ObjectPool.give(sectors[i]);
sectors.length = 0;
}
}
}
```

File 553: igw\com\service\server\outgoing\TransactionRequest.as

```
package com.service.server.outgoing
{
import com.enum.server.PurchaseTypeEnum;
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.ITransactionRequest;
import com.service.server.outgoing.starbase.StarbaseTransactionExpectedCost;

import flash.utils.ByteArray;

public class TransactionRequest implements ITransactionRequest
{
public var expectedCost:StarbaseTransactionExpectedCost = new
StarbaseTransactionExpectedCost();
public var purchaseType:int = PurchaseTypeEnum.INSTANT;

protected var _token:int;

protected var _protocolID:int;
protected var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(_token);
}

public
```

```
function writeExpectedCosts( output:ByteArray ):void { expectedCost.write(purchaseType,
output); }

public function get token():int { return _token; }
public function set token( v:int ):void { _token = v; }

public function destroy():void
{
expectedCost.destroy();
}
}
}

-----------------------------------
File 554: igw\com\service\server\outgoing\alliance\AllianceCreateAllianceRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceCreateAllianceRequest implements IRequest
{
public var name:String;
public var publicAlliance:Boolean;
public var description:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(name);
output.writeBoolean(publicAlliance);
output.writeUTF(description);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
function destroy():void
{
}
}
}
```

File 555: igw\com\service\server\outgoing\alliance\AllianceDemoteRequest.as

```
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceDemoteRequest implements IRequest
{
public var playerKey:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(playerKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 556: igw\com\service\server\outgoing\alliance\AllianceIgnoreInvitesRequest.as

```
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import
```

```
com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceIgnoreInvitesRequest implements IRequest
{
public var ignoreInvites:Boolean;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeBoolean(ignoreInvites);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 557: igw\com\service\server\outgoing\alliance\AllianceJoinRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceJoinRequest implements IRequest
{
public var allianceKey:String;
private var _protocolID:int;
private var _header:int;

public
```

```
function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(allianceKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 558: igw\com\service\server\outgoing\alliance\AllianceKickRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceKickRequest implements IRequest
{
public var playerKey:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
```

```
output.writeUTF(playerKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 559: igw\com\service\server\outgoing\alliance\AllianceLeaveRequest.as

```
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceLeaveRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 560: igw\com\service\server\outgoing\alliance\AlliancePromoteRequest.as

```
package
```

```
com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AlliancePromoteRequest implements IRequest
{
public var playerKey:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(playerKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 561: igw\com\service\server\outgoing\alliance\AllianceRequestBaselineRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceRequestBaselineRequest implements IRequest
{
public var allianceKey:String;
private
```

```
var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(allianceKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 562: igw\com\service\server\outgoing\alliance\AllianceRequestPublicsRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceRequestPublicsRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public var faction:String;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
```

```
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(faction);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 563: igw\com\service\server\outgoing\alliance\AllianceRequestRosterRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceRequestRosterRequest implements IRequest
{
public var allianceKey:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(allianceKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
```

```
}
}
}

-----------------------------------
File 564: igw\com\service\server\outgoing\alliance\AllianceSendInviteRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceSendInviteRequest implements IRequest
{
public var playerKey:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(playerKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 565: igw\com\service\server\outgoing\alliance\AllianceSetDescriptionRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import
```

```
com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceSetDescriptionRequest implements IRequest
{
public var description:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(description);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 566: igw\com\service\server\outgoing\alliance\AllianceSetMOTDRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceSetMOTDRequest implements IRequest
{
public var motd:String;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
```

```
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(motd);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 567: igw\com\service\server\outgoing\alliance\AllianceSetPublicRequest.as
package com.service.server.outgoing.alliance
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AllianceSetPublicRequest implements IRequest
{
public var publicAlliance:Boolean;
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
```

```
output.writeBoolean(publicAlliance);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 568: igw\com\service\server\outgoing\battle\BattleAttackOrderRequest.as
package com.service.server.outgoing.battle
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattleAttackOrderRequest implements IRequest
{
public var entityID:String;
public var targetID:String;
public var issuedTick:int;
public var subSystemTarget:int;
public var moveToTarget:Boolean = false;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(entityID);
output.writeInt(issuedTick);
output.writeUTF(targetID);
output.writeInt(subSystemTarget);
output.writeBoolean(moveToTarget);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
```

```
}

public function destroy():void
{
moveToTarget = false;
}
}
}

-----------------------------------
File 569: igw\com\service\server\outgoing\battle\BattleMoveOrderRequest.as
package com.service.server.outgoing.battle
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattleMoveOrderRequest implements IRequest
{
public var entityID:String;
public var startTick:int;
public var targetX:int;
public var targetY:int;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(entityID);
output.writeInt(startTick);
output.writeDouble(targetX);
output.writeDouble(targetY);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
function destroy():void
{
}
}
}
```

File 570: igw\com\service\server\outgoing\battle\BattlePauseRequest.as

```
package com.service.server.outgoing.battle
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattlePauseRequest implements IRequest
{
public var pause:Boolean;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeBoolean(pause);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 571: igw\com\service\server\outgoing\battle\BattleRetreatRequest.as

```
package com.service.server.outgoing.battle
{
import
```

```
com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattleRetreatRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
// additional data goes here
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 572: igw\com\service\server\outgoing\battle\BattleToggleModuleOrderRequest.as
package com.service.server.outgoing.battle
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattleToggleModuleOrderRequest implements IRequest
{
public var entityID:String;
public var issuedTick:int;
public var targetID:String;
public var moduleID:int;

private
```

```
var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(entityID);
output.writeInt(issuedTick);
output.writeUTF(targetID);
output.writeInt(moduleID);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 573: igw\com\service\server\outgoing\battlelog\BattleLogDetailRequest.as
package com.service.server.outgoing.battlelog
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattleLogDetailRequest implements IRequest
{
public var battleLogID:String;
private var _protocolID:int;
private var _header:int;

public function init(protocolID:int, header:int):void
{
_protocolID = protocolID;
_header = header;
}

public
```

```
function write(output:ByteArray):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(0);
output.writeUTF(battleLogID);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 574: igw\com\service\server\outgoing\battlelog\BattleLogListRequest.as

```
package com.service.server.outgoing.battlelog
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class BattleLogListRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init(protocolID:int, header:int):void
{
_protocolID = protocolID;
_header = header;
}

public function write(output:ByteArray):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(0);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
```

```
}
}
}
```

File 575: igw\com\service\server\outgoing\chat\ChatChangeRoomRequest.as

```
package com.service.server.outgoing.chat
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ChatChangeRoomRequest implements IRequest
{
public var channel:int;
public var roomKey:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(channel);
output.writeUTF(roomKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 576: igw\com\service\server\outgoing\chat\ChatIgnoreChatRequest.as

```
package com.service.server.outgoing.chat
{
import
```

```
com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ChatIgnoreChatRequest implements IRequest
{
public var playerKey:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(playerKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 577: igw\com\service\server\outgoing\chat\ChatReportChatRequest.as
package com.service.server.outgoing.chat
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ChatReportChatRequest implements IRequest
{
public var playerKey:String;

private
```

```
var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(playerKey);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 578: igw\com\service\server\outgoing\chat\ChatSendChatRequest.as
package com.service.server.outgoing.chat
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ChatSendChatRequest implements IRequest
{
public var channel:int;
public var playerKey:String;
public var message:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public
```

```
function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(channel);
output.writeUTF(playerKey);
output.writeUTF(message);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 579: igw\com\service\server\outgoing\leaderboard\LeaderboardRequest.as
package com.service.server.outgoing.leaderboard
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class LeaderboardRequest implements IRequest
{
public var type:int;
public var scope:int;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(type);
```

```
output.writeInt(scope);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 580:
igw\com\service\server\outgoing\leaderboard\LeaderboardRequestPlayerProfileRequest.as
package com.service.server.outgoing.leaderboard
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class LeaderboardRequestPlayerProfileRequest implements IRequest
{
public var playerKey:String;
public var nameSearch:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(playerKey);
output.writeUTF(nameSearch);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
```

```
}
}

-----------------------------------
File 581: igw\com\service\server\outgoing\mail\MailDeleteMailRequest.as
package com.service.server.outgoing.mail
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class MailDeleteMailRequest implements IRequest
{
public var mail:Vector.<String>;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);

output.writeUnsignedInt( mail.length );
for ( var idx:int = 0; idx < mail.length; ++idx )
{
output.writeUTF(mail[idx]);
}

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
```

File 582: igw\com\service\server\outgoing\mail\MailReadMailRequest.as

```
package com.service.server.outgoing.mail
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class MailReadMailRequest implements IRequest
{
public var mail:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(mail);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 583: igw\com\service\server\outgoing\mail\MailRequestInboxRequest.as

```
package com.service.server.outgoing.mail
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public
```

```
class MailRequestInboxRequest implements IRequest
{

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 584: igw\com\service\server\outgoing\mail\MailSendAllianceMailRequest.as
package com.service.server.outgoing.mail
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class MailSendAllianceMailRequest implements IRequest
{
public var subject:String;
public var body:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}
```

```
public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(subject);
output.writeUTF(body);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

------------------------------------
File 585: igw\com\service\server\outgoing\mail\MailSendMailRequest.as
package com.service.server.outgoing.mail
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class MailSendMailRequest implements IRequest
{
public var recipient:String;
public var subject:String;
public var body:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
```

```
output.writeUTF(recipient);
output.writeUTF(subject);
output.writeUTF(body);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 586: igw\com\service\server\outgoing\proxy\AuthRequest.as
package com.service.server.outgoing.proxy
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class AuthRequest implements IRequest
{
public var name:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(name);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
```

```
}

------------------------------------
File 587: igw\com\service\server\outgoing\proxy\ClientLoginRequest.as
package com.service.server.outgoing.proxy
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ClientLoginRequest implements IRequest
{
public var name:String;
public var challengeToken:String;
public var clientVersion:int;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(name);
output.writeUTF(challengeToken);
output.writeInt(clientVersion);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

------------------------------------
File 588: igw\com\service\server\outgoing\proxy\ProxyConnectToBattleRequest.as
package com.service.server.outgoing.proxy
{
```

```
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ProxyConnectToBattleRequest implements IRequest
{
public var key:String = "";

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(key);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 589: igw\com\service\server\outgoing\proxy\ProxyConnectToSectorRequest.as
package com.service.server.outgoing.proxy
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ProxyConnectToSectorRequest implements IRequest
{
public var key:String = "";

private
```

```
var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(key);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 590: igw\com\service\server\outgoing\proxy\ProxyReportCrashRequest.as
package com.service.server.outgoing.proxy
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ProxyReportCrashRequest implements IRequest
{
public var dataStr:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
```

```
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(dataStr);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 591: igw\com\service\server\outgoing\proxy\ProxyReportLoginDataRequest.as

```
package com.service.server.outgoing.proxy
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ProxyReportLoginDataRequest implements IRequest
{
public var dataStr:String;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(dataStr);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
function destroy():void
{
}
}
}
```

File 592: igw\com\service\server\outgoing\proxy\ProxyTutorialStepCompletedMessage.as

```
package com.service.server.outgoing.proxy
{
import com.Application;
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class ProxyTutorialStepCompletedMessage implements IRequest
{
private var _protocolID:int;
private var _header:int;

public var stepId:int;
public var kabamNaid:String;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(stepId);
output.writeUTF(kabamNaid);
output.writeInt(Application.NETWORK);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 593: igw\com\service\server\outgoing\proxy\TimeSyncRequest.as

```
package com.service.server.outgoing.proxy
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;
import flash.utils.getTimer;

public class TimeSyncRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(getTimer());
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}
```

File 594: igw\com\service\server\outgoing\sector\SectorOrderRequest.as

```
package com.service.server.outgoing.sector
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class SectorOrderRequest implements IRequest
{
```

```
public var entityId:String = '';
public var orderType:int;
public var targetLocationX:int;
public var targetLocationY:int;
public var targetId:String = '';
public var destinationSector:String = '';

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(entityId);
output.writeInt(orderType);
output.writeDouble(targetLocationX);
output.writeDouble(targetLocationY);
output.writeUTF(targetId);
output.writeUTF(destinationSector);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 595: igw\com\service\server\outgoing\sector\SectorRequestBaselineRequest.as
package com.service.server.outgoing.sector
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class SectorRequestBaselineRequest implements IRequest
{
```

```
public var viewX:int;
public var viewY:int;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(viewX);
output.writeInt(viewY);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-------------------------------------
File 596: igw\com\service\server\outgoing\sector\SectorSetViewLocationRequest.as
package com.service.server.outgoing.sector
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class SectorSetViewLocationRequest implements IRequest
{
public var x:int;
public var y:int;

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID
```

```
= protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(x);
output.writeInt(y);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{

}
}
}

-------------------------------------
File 597: igw\com\service\server\outgoing\starbase\StarbaseAllScoresRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseAllScoresRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
```

```
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }


}
}
```

File 598: igw\com\service\server\outgoing\starbase\StarbaseBookmarkDeleteRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBookmarkDeleteRequest extends TransactionRequest
{
public var index:int;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUnsignedInt(index);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 599: igw\com\service\server\outgoing\starbase\StarbaseBookmarkSaveRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public
```

```
class StarbaseBookmarkSaveRequest extends TransactionRequest
{
public var name:String;
public var sector:String;
public var nameProto:String;
public var enumProto:String;
public var sectorProto:String;
public var x:int;
public var y:int;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(name);
output.writeUTF(sector);
output.writeUTF(nameProto);
output.writeUTF(enumProto);
output.writeUTF(sectorProto);
output.writeInt(x);
output.writeInt(y);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-------------------------------------
File 600: igw\com\service\server\outgoing\starbase\StarbaseBookmarkUpdateRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.model.player.BookmarkVO;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBookmarkUpdateRequest extends TransactionRequest
{
public var bookmark:BookmarkVO;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUnsignedInt(bookmark.index);
output.writeUTF(bookmark.name);
output.writeUTF(bookmark.sector);
output.writeUTF(bookmark.sectorNamePrototype.name);
```

```
output.writeUTF(bookmark.sectorEnumPrototype.name);
output.writeUTF(bookmark.sectorPrototype.name);
output.writeInt(bookmark.x);
output.writeInt(bookmark.y);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 601: igw\com\service\server\outgoing\starbase\StarbaseBribeContractRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBribeContractRequest extends TransactionRequest
{
public var tradeRoutePersistence:String;
public var pointsBought:int;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(tradeRoutePersistence);
output.writeInt(pointsBought);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 602: igw\com\service\server\outgoing\starbase\StarbaseBuildNewBuildingRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import
```

```
com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBuildNewBuildingRequest extends TransactionRequest
{
public var buildingPrototype:String;
public var locationX:int;
public var locationY:int;
public var centerSpaceBase:Boolean; // set this true if building in your centerSpace base

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(buildingPrototype);
output.writeInt(locationX);
output.writeInt(locationY);
output.writeBoolean(centerSpaceBase);
output.writeInt(purchaseType);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-------------------------------------
File 603: igw\com\service\server\outgoing\starbase\StarbaseBuildShipRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;
import flash.utils.Dictionary;

public class StarbaseBuildShipRequest extends TransactionRequest
{
public var modules:Dictionary;
public var shipPrototype:String;
public var shipName:String;
public var slots:Array;
public var centerSpaceBase:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
```

```
output.writeUTF(shipPrototype);
//output.writeUTF(shipName);
output.writeInt(purchaseType);
output.writeUnsignedInt(slots.length);
for (var i:int = 0; i < slots.length; i++)
{
output.writeUTF(slots[i]);
if (slots[i] in modules)
{
if (modules[slots[i]] != null)
output.writeUTF(modules[slots[i]].name);
else
output.writeUTF('');
} else
output.writeUTF('');
}
output.writeBoolean(centerSpaceBase);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
modules = null;
slots = null;
}
}
}

-----------------------------------
File 604: igw\com\service\server\outgoing\starbase\StarbaseBuyOtherStoreItemRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBuyOtherStoreItemRequest extends TransactionRequest
{
public var storeItemPrototype:String;
public var itemType:String;
public var itemAmount:int;
public var centerSpaceBase:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(storeItemPrototype);
```

```
output.writeUTF(itemType);
output.writeInt(itemAmount);
output.writeBoolean(centerSpaceBase);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 605: igw\com\service\server\outgoing\starbase\StarbaseBuyoutBlueprintRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBuyoutBlueprintRequest extends TransactionRequest
{
public var blueprintPersistence:String;
public var partsPurchased:int;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(blueprintPersistence);
writeExpectedCosts(output);
output.writeInt(partsPurchased);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 606: igw\com\service\server\outgoing\starbase\StarbaseBuyResourceRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import
```

```
com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBuyResourceRequest extends TransactionRequest
{
public var resource:String;
public var percent:int;
public var centerSpaceBase:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(resource);
output.writeInt(percent);
output.writeBoolean(centerSpaceBase);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-------------------------------------
File 607: igw\com\service\server\outgoing\starbase\StarbaseBuyStoreItemRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseBuyStoreItemRequest extends TransactionRequest
{
public var buffPrototype:String;
public var centerSpaceBase:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(buffPrototype);
output.writeBoolean(centerSpaceBase);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
override function destroy():void
{
super.destroy();
}
}
}

------------------------------------
File 608: igw\com\service\server\outgoing\starbase\StarbaseCancelContractRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseCancelContractRequest extends TransactionRequest
{
public var tradeRoutePersistence:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(tradeRoutePersistence);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

------------------------------------
File 609: igw\com\service\server\outgoing\starbase\StarbaseCancelTransactionRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseCancelTransactionRequest extends TransactionRequest
{
public var serverKey:String;
public var expectedRefund:StarbaseTransactionExpectedCost = new
StarbaseTransactionExpectedCost;

public
```

```
override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(serverKey);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}
```

File 610:
igw\com\service\server\outgoing\starbase\StarbaseClaimAchievementRewardRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseClaimAchievementRewardRequest implements IRequest
{
public var achievement:String; // persistence DB key of the achievement you are claiming [not the prototype key]

private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(achievement);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }


}
}
-----------------------------------
File 611: igw\com\service\server\outgoing\starbase\StarbaseClaimDailyRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseClaimDailyRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }


}
```

```
}
```

File 612:
igw\com\service\server\outgoing\starbase\StarbaseCompleteBlueprintResearchRequest.as
```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseCompleteBlueprintResearchRequest extends TransactionRequest
{
public var blueprintPersistence:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(blueprintPersistence);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 613: igw\com\service\server\outgoing\starbase\StarbaseExtendContractRequest.as
```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseExtendContractRequest extends TransactionRequest
{
public var tradeRoutePersistence:String;
public var extensionHours:Number;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(tradeRoutePersistence);
output.writeDouble(extensionHours);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}
```

```
public override function destroy():void
{
super.destroy();
}
}
}
```

File 614: igw\com\service\server\outgoing\starbase\StarbaseGetPaywallPayoutsRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseGetPaywallPayoutsRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }


}
}
```

```
------------------------------------
File 615: igw\com\service\server\outgoing\starbase\StarbaseInstancedMissionStartRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseInstancedMissionStartRequest extends TransactionRequest
{
public var instancedMissionID:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(instancedMissionID);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

------------------------------------
File 616: igw\com\service\server\outgoing\starbase\StarbaseLaunchFleetRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseLaunchFleetRequest extends TransactionRequest
{
public var fleets:Array;

public override function write( output:ByteArray ):void
{
super.write(output);
var len:uint = fleets.length;
output.writeUnsignedInt(len);
for (var i:uint = 0; i < len; ++i)
output.writeUTF(fleets[i].id);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
override function destroy():void
{
}
}
}
```

File 617: igw\com\service\server\outgoing\starbase\StarbaseMintNFTRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMintNFTRequest extends TransactionRequest
{
public var tokenType:int;
public var tokenAmount:int;
public var tokenPrototype:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeInt(tokenType);
output.writeInt(tokenAmount);
output.writeUTF(tokenPrototype);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 618: igw\com\service\server\outgoing\starbase\StarbaseMissionAcceptRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMissionAcceptRequest extends TransactionRequest
{
public var missionPersistence:String;

public override function write( output:ByteArray ):void
{
```

```
super.write(output);
output.writeUTF(missionPersistence);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

-----------------------------------
File 619: igw\com\service\server\outgoing\starbase\StarbaseMissionAcceptRewardsRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMissionAcceptRewardsRequest extends TransactionRequest
{
public var missionPersistence:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(missionPersistence);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

-----------------------------------
File 620: igw\com\service\server\outgoing\starbase\StarbaseMissionStepRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMissionStepRequest extends TransactionRequest
{
public var missionPrototype:String;
public
```

```
var progress:int;


public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(missionPrototype);
output.writeInt(progress);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

-----------------------------------
File 621: igw\com\service\server\outgoing\starbase\StarbaseMotDReadRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMotDReadRequest extends TransactionRequest
{
public var key:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(key);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
-----------------------------------
File 622: igw\com\service\server\outgoing\starbase\StarbaseMoveBuildingRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import
```

```
flash.utils.ByteArray;

public class StarbaseMoveBuildingRequest extends TransactionRequest
{
public var buildingKey:String;
public var locationX:int;
public var locationY:int;
public var centerSpaceBase:Boolean; // set this true if building in your centerSpace base

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(buildingKey);
output.writeInt(locationX);
output.writeInt(locationY);
output.writeBoolean(centerSpaceBase);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

-----------------------------------
File 623: igw\com\service\server\outgoing\starbase\StarbaseMoveStarbaseRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMoveStarbaseRequest extends TransactionRequest
{
public var targetPlayer:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(targetPlayer);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
```

```
}

------------------------------------
File 624:
igw\com\service\server\outgoing\starbase\StarbaseMoveStarbaseToTransgateRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseMoveStarbaseToTransgateRequest extends TransactionRequest
{
public var targetSector:String;
public var targetTransgate:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(targetSector);
output.writeUTF(targetTransgate);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

------------------------------------
File 625: igw\com\service\server\outgoing\starbase\StarbaseNegotiateContractRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseNegotiateContractRequest extends TransactionRequest
{
public var centerSpaceBase:Boolean;
public var contractPrototype:String;
public var factionPrototype:String;
public var productivity:Number;
public var payout:Number;
public
```

```
var duration:Number;
public var frequency:Number;
public var security:Number;
public var fullBribe:Boolean;
public var accepted:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeBoolean(centerSpaceBase);
output.writeUTF(contractPrototype);
output.writeUTF(factionPrototype);
output.writeDouble(productivity);
output.writeDouble(payout);
output.writeDouble(duration);
output.writeDouble(frequency);
output.writeDouble(security);
output.writeBoolean(fullBribe);
writeExpectedCosts(output);
output.writeBoolean(accepted);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 626: igw\com\service\server\outgoing\starbase\StarbaseRecallFleetRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRecallFleetRequest extends TransactionRequest
{
public var fleet:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(fleet);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
override function destroy():void
{
}
}
}
```

File 627: igw\com\service\server\outgoing\starbase\StarbaseRecycleBuildingRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRecycleBuildingRequest extends TransactionRequest
{
public var buildingPersistence:String;
public var expectedRefund:StarbaseTransactionExpectedCost = new
StarbaseTransactionExpectedCost;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(buildingPersistence);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}
```

File 628: igw\com\service\server\outgoing\starbase\StarbaseRecycleShipRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRecycleShipRequest extends TransactionRequest
{
public var shipPersistence:String;
public var expectedRefund:StarbaseTransactionExpectedCost = new
StarbaseTransactionExpectedCost;

public
```

```
override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(shipPersistence);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

-----------------------------------
File 629: igw\com\service\server\outgoing\starbase\StarbaseRefitBuildingRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;
import flash.utils.Dictionary;

public class StarbaseRefitBuildingRequest extends TransactionRequest
{
public var buildingPersistence:String;
public var modules:Dictionary;
public var slots:Array;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(buildingPersistence);
output.writeInt(purchaseType);
// NOTE - this is stolen from StarbaseBuildShipRequest, keep them in sync!
output.writeUnsignedInt(slots.length);
for (var i:int = 0; i < slots.length; i++)
{
output.writeUTF(slots[i]);
if (modules.hasOwnProperty(slots[i]))
output.writeUTF(modules[slots[i]] != null ? modules[slots[i]].name : '');
else
output.writeUTF('');
}
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
```

```
super.destroy();
modules = null;
slots = null;
}
}
}

-----------------------------------
File 630: igw\com\service\server\outgoing\starbase\StarbaseRefitShipRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;
import flash.utils.Dictionary;

import org.adobe.utils.DictionaryUtil;

public class StarbaseRefitShipRequest extends TransactionRequest
{
public var shipPersistence:String;
public var modules:Dictionary;
public var shipName:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(shipPersistence);
output.writeUTF(shipName);
output.writeInt(purchaseType);
// NOTE - this is stolen from StarbaseBuildShipRequest, keep them in sync!
output.writeUnsignedInt(DictionaryUtil.getLength(modules));

for (var key:String in modules)
{
output.writeUTF(key);
if (modules[key] != null)
output.writeUTF(modules[key].name);
else
output.writeUTF('');
}

writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
```

```
modules = null;
}
}
}

------------------------------------
File 631: igw\com\service\server\outgoing\starbase\StarbaseRenameFleetRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRenameFleetRequest extends TransactionRequest
{
public var fleetId:String;
public var newName:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(fleetId);
output.writeUTF(newName);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
}
}

------------------------------------
File 632: igw\com\service\server\outgoing\starbase\StarbaseRenamePlayerRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRenamePlayerRequest extends TransactionRequest
{
public var newName:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(newName);
```

```
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 633: igw\com\service\server\outgoing\starbase\StarbaseRepairBaseRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRepairBaseRequest extends TransactionRequest
{
public var centerSpaceBase:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeInt(purchaseType);
output.writeBoolean(centerSpaceBase);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 634: igw\com\service\server\outgoing\starbase\StarbaseRepairFleetRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRepairFleetRequest extends TransactionRequest
{
```

```
public var fleetId:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(fleetId);
output.writeInt(purchaseType);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 635: igw\com\service\server\outgoing\starbase\StarbaseRequestAchievementsRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseRequestAchievementsRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }


}
}
-----------------------------------
File 636: igw\com\service\server\outgoing\starbase\StarbaseRerollBlueprintChanceRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRerollBlueprintChanceRequest extends TransactionRequest
{
public var battleKey:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(battleKey);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 637: igw\com\service\server\outgoing\starbase\StarbaseRerollBlueprintReceivedRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseRerollBlueprintReceivedRequest extends TransactionRequest
{
public
```

```
var battleKey:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(battleKey);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 638: igw\com\service\server\outgoing\starbase\StarbaseResearchRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseResearchRequest extends TransactionRequest
{
public var researchPrototype:String;
public var centerSpaceBase:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(researchPrototype);
output.writeInt(purchaseType);
output.writeBoolean(centerSpaceBase);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 639: igw\com\service\server\outgoing\starbase\StarbaseResecureContractRequest.as

```
package
```

```
com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseResecureContractRequest extends TransactionRequest
{
public var tradeRoutePersistence:String;
public var efficiencySecured:Number;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(tradeRoutePersistence);
output.writeDouble(efficiencySecured);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 640: igw\com\service\server\outgoing\starbase\StarbaseSetClientSettingsRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseSetClientSettingsRequest extends TransactionRequest
{
public var settings:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(settings);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
}
```

```
}
}

-----------------------------------
File 641: igw\com\service\server\outgoing\starbase\StarbaseSkipTrainingRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseSkipTrainingRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeInt(0); // transaction token
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }


}
}
-----------------------------------
File 642: igw\com\service\server\outgoing\starbase\StarbaseSpeedUpTransactionRequest.as
package com.service.server.outgoing.starbase
{
```

```
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseSpeedUpTransactionRequest extends TransactionRequest
{
public var serverKey:String;
public var milliseconds:int;
public var fromStore:Boolean;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(serverKey);
output.writeInt(purchaseType);
output.writeInt(milliseconds);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 643: igw\com\service\server\outgoing\starbase\StarbaseTransactionExpectedCost.as
package com.service.server.outgoing.starbase
{
import flash.utils.ByteArray;

public class StarbaseTransactionExpectedCost
{
public var time_cost_milliseconds:int;
public var alloyCost:int;
public var syntheticCost:int;
public var energyCost:int;
public var creditsCost:int;
public var hardCurrencyCost:int;

public function write( purchaseType:int, output:ByteArray ):void
{
output.writeInt(time_cost_milliseconds);
output.writeInt(alloyCost);
output.writeInt(syntheticCost);
output.writeInt(energyCost);
output.writeInt(creditsCost);
```

```
output.writeInt(hardCurrencyCost);
}

public function destroy():void
{
time_cost_milliseconds = 0;
alloyCost = 0;
syntheticCost = 0;
energyCost = 0;
creditsCost = 0;
hardCurrencyCost = 0;
}
}
}

-------------------------------------
File 644: igw\com\service\server\outgoing\starbase\StarbaseUpdateFleetRequest.as
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseUpdateFleetRequest extends TransactionRequest
{
public var fleet:FleetVO;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(fleet.id);
var ships:Vector.<ShipVO> = fleet.ships;
var len:uint = ships.length;
output.writeUnsignedInt(len);
for (var i:uint = 0; i < len; ++i)
{
if (ships[i])
{
output.writeUTF(ships[i].id);
} else
{
output.writeUTF("");
}
}
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public
```

```
override function destroy():void
{
}
}
}
```

File 645: igw\com\service\server\outgoing\starbase\StarbaseUpgradeBuildingRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.service.server.outgoing.TransactionRequest;

import flash.utils.ByteArray;

public class StarbaseUpgradeBuildingRequest extends TransactionRequest
{
public var buildingPersistence:String;

public override function write( output:ByteArray ):void
{
super.write(output);
output.writeUTF(buildingPersistence);
output.writeInt(purchaseType);
writeExpectedCosts(output);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public override function destroy():void
{
super.destroy();
}
}
}
```

File 646: igw\com\service\server\outgoing\starbase\StarbaseVerifyPaymentRequest.as

```
package com.service.server.outgoing.starbase
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class StarbaseVerifyPaymentRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public
```

```
var externalTrkid:String;
public var payoutId:String;
public var responseData:String;
public var responseSignature:String;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(externalTrkid);
output.writeUTF(payoutId);
output.writeUTF(responseData);
output.writeUTF(responseSignature);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}

public function set protocolID(value:int):void { _protocolID = value; }

public function set header(value:int):void { _header = value; }



}
}
-------------------------------------
File 647: igw\com\service\server\outgoing\universe\UniverseCreateCharacterRequest.as
package com.service.server.outgoing.universe
{
import com.enum.server.RequestEnum;
import com.enum.server.SpeakerEnum;
import com.service.server.IRequest;

import flash.utils.ByteArray;

public class UniverseCreateCharacterRequest implements IRequest
{
private var _protocolID:int;
private var _header:int;

public
```

```
var factionPrototype:String;
public var racePrototype:String;
public var avatar:String;
public var name:String;

public function init( protocolID:int, header:int ):void
{
_protocolID = protocolID;
_header = header;
}

public function write( output:ByteArray ):void
{
output.writeByte(_protocolID);
output.writeByte(SpeakerEnum.CLIENT_SPEAKER);
output.writeByte(_header);

output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_START);
output.writeUTF(factionPrototype);
output.writeUTF(racePrototype);
output.writeUTF(name);
output.writeInt(RequestEnum.BINARY_OBJECT_SEQUENCE_TOKEN_END);
}

public function destroy():void
{
}
}
}

-----------------------------------
File 648: igw\com\service\server\replicable\IReplicable.as
package com.service.server.replicable
{
import com.service.server.BinaryInputStream;

public interface IReplicable
{
function decode( input:BinaryInputStream ):int;
function read( input:BinaryInputStream ):void;
function resetDeltas( ):void;
}
}
-----------------------------------
File 649: igw\com\service\server\replicable\ReplicableMap.as
package com.service.server.replicable
{
import com.enum.server.ReplicableOpEnum;
import com.service.server.BinaryInputStream;

import
```

```
flash.utils.Dictionary;

import org.shared.ObjectPool;

public dynamic class ReplicableMap extends Dictionary implements IReplicable
{
public function readKey( input:BinaryInputStream ):* { return null; }
public function readRemove( input:BinaryInputStream ):* { return null; }
public function createValue():* { return null; }

public var added:Dictionary = new Dictionary;
public var modified:Dictionary = new Dictionary;
public var removed:Array = new Array;

protected var _count:int;
protected var _i:int;

public function resetDeltas():void
{
for each(var rm:IReplicable in modified)
{
rm.resetDeltas();
}
added = new Dictionary;
modified = new Dictionary;

removed.length = 0;
}

public function read( input:BinaryInputStream ):void
{
_count = input.readUnsignedInt();
for (var id:* in this)
{
delete this[id];
}

for (_i = 0; _i < _count; ++_i)
{
var key:* = readKey(input);
var element:* = createValue();
element.read(input);
this[key] = element;
}
}

public function decode( input:BinaryInputStream ):int
{
for (; ; )
{
var
```

```
operation:int = input.readByte();
if (operation < 0)
{
return operation;
}

switch (operation)
{
case ReplicableOpEnum.Modifychild:
{
var key:* = readKey(input);
var found1:IReplicable = this[key];
if (found1 != null)
{
var level:int = 1 + found1.decode(input);
modified[key] = found1;
if (level < 0)
{
return level;
}
} else
{
if( key != null)
throw new Error("ReplicableSet modifying " + key + " that is not present");
}
break;
}

case ReplicableOpEnum.Insertdefault:
{
var insertdefaultkey:* = readKey(input);
this[insertdefaultkey] = createValue();
break;
}

case ReplicableOpEnum.Set:
{
var setKey:* = readKey(input);
var objSet:* = createValue();
objSet.read(input);
this[setKey] = objSet;
added[setKey] = objSet;
break;
}

case ReplicableOpEnum.Erase:
{
var payload:* = readRemove(input);
delete this[payload.id];

removed.push(payload);
```

```
break;
}

case ReplicableOpEnum.Copy:
{
read(input);
break;
}

case ReplicableOpEnum.Clear:
{
this.length = 0;
break;
}

default:
{
throw new Error("ReplicableSet invalid operation " + String(operation));
break;
}
}
}
return 0;
}
}
}

------------------------------------
File 650: igw\com\service\server\replicable\ReplicableSet.as
package com.service.server.replicable
{
import com.enum.server.ReplicableOpEnum;

import org.shared.ObjectPool;
import com.service.server.BinaryInputStream;

public dynamic class ReplicableSet extends Array implements IReplicable
{
public var readKey:Function;
public var readRemove:Function;
public var elementType:Class;

public var added:Array = new Array;
public var modified:Array = new Array;
public var removed:Array = new Array;

protected var _count:int;
protected var _i:int;

public
```

```
function resetDeltas():void
{
modified.length = 0;
added.length = 0;
removed.length = 0;
}

public function find( key:* ):*
{
for (_i = 0; _i < this.length; _i++)
{
if (this[_i].id == key)
{
return this[_i];
}
}
return null;
}

private function erase( key:* ):Boolean
{
for (_i = 0; _i < this.length; _i++)
{
if (this[_i].id == key)
{
this.splice(_i, 1);
return true;
}
}
return false;
}

public function read( input:BinaryInputStream ):void
{
_count = input.readUnsignedInt();
length = 0;
for (_i = 0; _i < _count; ++_i)
{
var element:* = ObjectPool.get(elementType);
element.read(input);
super.push(element);
}
}

public function decode( input:BinaryInputStream ):int
{
for (; ; )
{
var operation:int = input.readByte();
if (operation < 0)
{
```

```
return operation;
}

switch (operation)
{
case ReplicableOpEnum.Modifychild:
{
var key:* = readKey(input);
var found1:IReplicable = find(key);
if (found1 != null)
{
var level:int = 1 + found1.decode(input);
if (modified.indexOf(found1) < 0)
{
modified.push(found1);
}
if (level < 0)
{
return level;
}
} else
{
throw new Error("ReplicableSet modifying " + key + " that is not present");
}
break;
}

case ReplicableOpEnum.Set:
{
var objSet:* = ObjectPool.get(elementType);
objSet.read(input);
erase(objSet.id); // erase the old one if it exists
this.push(objSet); // add the new one
added.push(objSet);
break;
}

case ReplicableOpEnum.Erase:
{
var payload:* = readRemove(input);
var erased:Boolean = erase(payload.id);
if (!erased)
{
throw new Error("ReplicableSet erasing " + payload.id + " that is not present");
}

removed.push(payload);
break;
}

case
```

```
ReplicableOpEnum.Copy:
{
read(input);
break;
}

case ReplicableOpEnum.Clear:
{
this.length = 0;
break;
}

default:
{
throw new Error("ReplicableSet invalid operation " + String(operation));
break;
}
}
}
return 0;
}
}
}

------------------------------------
File 651: igw\com\service\server\replicable\ReplicableStruct.as
package com.service.server.replicable
{
import com.enum.server.ReplicableOpEnum;
import com.service.server.BinaryInputStream;

import flash.geom.Point;

public class ReplicableStruct implements IReplicable
{
public function read( input:BinaryInputStream ):void {}

public function readInt( input:BinaryInputStream ):int { return input.readInt(); }

public function readUnsignedByte( input:BinaryInputStream ):int { return
input.readUnsignedByte(); }

public function readUTF( input:BinaryInputStream ):String { return input.readUTF(); }

public function readShort( input:BinaryInputStream ):int { return input.readShort(); }

public function readBoolean( input:BinaryInputStream ):Boolean { return input.readBoolean(); }

public function readFloat( input:BinaryInputStream ):Number { return input.readFloat(); }

public
```

```
function readLocation( input:BinaryInputStream ):Point
{
var result:Point = new Point();
result.x = input.readFloat();
result.y = input.readFloat();
return result;
//p.setTo(input.readFloat(), input.readFloat());
}

public function resetDeltas():void
{

}

public function decode( input:BinaryInputStream ):int
{
while (true)
{
var index:int = input.readByte();
if (index < 0)
{
return index;
}
if (index == ReplicableOpEnum.Copy)
{
// special copy everything token
read(input);
} else
{
var propName:String = propertyNames[index];
var child:* = this[propName];
var level:int = 1;
if (child is IReplicable)
{
level = 1 + IReplicable(child).decode(input);
} else
{
this[propName] = this[propertyReaders[index]](input);
}

if (level < 0)
{
return level;
}
}
}
return 0;
}

public function get propertyNames():Vector.<String> { return null; }
public
```

```
function get propertyReaders():Vector.<String> { return null; }

}
}

-----------------------------------
File 652: igw\com\service\server\replicable\ReplicableVector.as
package com.service.server.replicable
{
import com.enum.server.ReplicableOpEnum;

import org.shared.ObjectPool;
import com.service.server.BinaryInputStream;

public dynamic class ReplicableVector extends Array implements IReplicable
{
public var added:Array = new Array;
public var elementType:Class;
public var modified:Array = new Array;
public var removed:Array = new Array;

public function resetDeltas():void
{
modified.length = 0;
added.length = 0;
removed.length = 0;
}

public function read( input:BinaryInputStream ):void
{
var count:int = input.readShort();
for (var i:int = 0; i < count; ++i)
{
var element:* = ObjectPool.get(elementType);
element.read(input);
super.push(element);
}
}

public function decode( input:BinaryInputStream ):int
{
for (; ; )
{
var operation:int = input.readByte();
if (operation < 0)
{
return operation;
}

switch (operation)
{
```

```
case ReplicableOpEnum.Modifychild:
{
var modifyIndex:uint = input.readUnsignedByte();
var modifyItem:IReplicable = this[modifyIndex];
var level:int = 1 + modifyItem.decode(input);
modified.push(modifyItem);
if (level < 0)
{
return level;
}
break;
}

case ReplicableOpEnum.Pushback:
{
var pushedElement:* = ObjectPool.get(elementType);
pushedElement.read(input);
this.push(pushedElement);
added.push(pushedElement);
break;
}

case ReplicableOpEnum.Set:
{
var setIndex:uint = input.readUnsignedByte();
this[setIndex].read(input);
break;
}

case ReplicableOpEnum.Erase:
{
var eraseIndex:uint = input.readUnsignedByte();
var erased:Array = super.splice(eraseIndex, 1);
removed.push(erased[0]);
break;
}

case ReplicableOpEnum.Copy:
{
length = 0;
var copycount:int = input.readUnsignedByte();
for (var i:int = 0; i < copycount; i++)
{
var copyelement:* = ObjectPool.get(elementType);
copyelement.read(input);
this.push(copyelement);
}
break;
}

case
```

```
ReplicableOpEnum.Clear:
{
this.length = 0;
break;
}

default:
{
throw new Error("ReplicableVector invalid operation " + String(operation));
break;
}

}
}
return 0;
}
}
}

-------------------------------------
File 653: igw\com\service\xsolla\XsollaAPI.as
package com.service.xsolla
{
import com.service.ExternalInterfaceAPI;

public class XsollaAPI
{
public function get gameAuthToken():String
{
return ExternalInterfaceAPI.getPlayerXsollaGameAuthToken();
}
}
}
-------------------------------------
File 654: igw\com\ui\GameView.as
package com.ui
{
import com.Application;
import com.enum.PositionEnum;
import com.presenter.shared.IUIPresenter;
import com.ui.core.View;
import com.ui.core.effects.EffectFactory;

import flash.events.Event;
import flash.geom.Rectangle;

import org.parade.enum.PlatformEnum;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.starling.core.Starling;

public
```

```
class GameView extends View
{
[PostConstruct]
override public function init():void
{
super.init();
presenter.removeStateListener(onStateChange);

addEffects();
effectsIN();
onResize();
}

private function onResize( e:Event = null ):void
{
if (Application.STARLING_ENABLED)
{
var viewPortRectangle:Rectangle = new Rectangle();
viewPortRectangle.width = DeviceMetrics.WIDTH_PIXELS;
viewPortRectangle.height = DeviceMetrics.HEIGHT_PIXELS;
Starling.current.viewPort = viewPortRectangle;

Starling.current.stage.stageWidth = DeviceMetrics.WIDTH_PIXELS;
Starling.current.stage.stageHeight = DeviceMetrics.HEIGHT_PIXELS;
}

var scaleX:Number = Math.min(1, DeviceMetrics.WIDTH_PIXELS /
Application.MIN_SCREEN_X);
var scaleY:Number = Math.min(1, DeviceMetrics.HEIGHT_PIXELS /
Application.MIN_SCREEN_Y);
Application.SCALE = Math.min(scaleX, scaleY);
if(CONFIG::IS_DESKTOP){
//Application.SCALE = .63;
Application.SCALE = Math.max(Application.SCALE, .63);
} else {
Application.SCALE = Math.max(Application.SCALE, DeviceMetrics.PLATFORM ==
PlatformEnum.BROWSER ? .63 : .1);
}
presenter.changeResolution();
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.TOP,
onResize));
}

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override
```

```
public function get type():String { return ViewEnum.UI; }
}
}

------------------------------------
File 655: igw\com\ui\PreloadView.as
ï»¿package com.ui
{
import com.Application;
import com.enum.TimeLogEnum;
import com.event.ServerEvent;
import com.model.player.CurrentUser;
import com.presenter.preload.IPreloadPresenter;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.effects.EffectFactory;
import com.ui.core.effects.GenericMoveEffect;
import com.util.TimeLog;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.net.SharedObject;

import org.parade.enum.ViewEnum;

public class PreloadView extends View
{
[Embed(source="LoadingScreen.png")]
public static var loadingScreen:Class;

private var _loadBar:ProgressBar;
private var _loadingImage:Bitmap;
private var _loginResponse:Object;

[PostConstruct]
override public function init():void
{
super.init();
TimeLog.startTimeLog(TimeLogEnum.PRELOAD_SCREEN);
_loadingImage = new loadingScreen();

_loadBar = new ProgressBar();
_loadBar.init(ProgressBar.HORIZONTAL, new Bitmap(new BitmapData(_loadingImage.width -
10, 30, false, 0x226622)), new Bitmap(new BitmapData(_loadingImage.width, 40, false,
0x222222)), 1);
_loadBar.setMinMax(0, 1);
_loadBar.amount = 0;

addChild(_loadingImage);
addChild(_loadBar);

_loginResponse
```

```
= root.loaderInfo.parameters.LoginResponse;

presenter.beginSignal.add(onBegin);
presenter.completeSignal.add(onComplete);
presenter.addLoadCompleteListener(onLoadComplete);
presenter.progressSignal.add(onProgress);
presenter.beginLoad();

addEffects();
effectsIN();

if(CONFIG::IS_MOBILE){
_loadingImage.visible = _loadBar.visible = false;
}
}

private function onBegin( total:int ):void
{
_loadBar.setMinMax(0, total + 1);
}

private function onProgress( loaded:int, total:int ):void
{
if (isNaN(loaded))
return;
_loadBar.amount = loaded;
}

private function onLoadComplete():void
{
presenter.removeLoadCompleteListener(onLoadComplete);

if (Application.CONNECTION_STATE == ServerEvent.NOT_CONNECTED &&
Application.NETWORK != Application.NETWORK_KONGREGATE)
{
var savedName:SharedObject = SharedObject.getLocal("playerID");
if (savedName.size > 0 && savedName.data.playerid != null)
{
showInputAlert('LOGIN', 'Please Enter Your Player Id', 'Accept', savePlayerInfo, null, null, null,
null, false, 20, savedName.data.playerid);
} else
{
showInputAlert('LOGIN', 'Please Enter Your Player Id', 'Accept', savePlayerInfo, null, null, null,
null, false, 20, '1');
}
} else
presenter.transitionToLoad();

_loadBar.visible = false;
_loadingImage.visible = false;
}
```

```
private function savePlayerInfo( id:String ):void
{
var savedName:SharedObject = SharedObject.getLocal("playerID");
savedName.data.playerid = id;
savedName.flush();
CurrentUser.naid = id;
presenter.transitionToLoad();
}

private function onResize():void
{
_loadingImage.x = 0;
_loadingImage.y = 0;

_loadBar.x = _loadingImage.x;
_loadBar.y = _loadingImage.y + _loadingImage.height + 10;
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.resizeEffect(onResize));
_effects.addEffect(EffectFactory.simpleBackingEffect(1, 0, 0));
_effects.addEffect(EffectFactory.genericMoveEffect(GenericMoveEffect.UP,
GenericMoveEffect.UP, .1, .1));
}
private function onComplete():void { destroy(); }

override public function get height():Number { return 400; }
override public function get width():Number { return 1024; }

[Inject]
public function set presenter( value:IPreloadPresenter ):void { _presenter = value; }
public function get presenter():IPreloadPresenter { return IPreloadPresenter(_presenter); }

override public function get type():String { return ViewEnum.UI; }

override public function destroy():void
{
TimeLog.endTimeLog(TimeLogEnum.PRELOAD_SCREEN);
super.destroy();

_loadBar.destroy();
_loadBar = null;

_loadingImage.bitmapData.dispose();
_loadingImage = null;
}
}
}
```

File 656: igw\com\ui\ReconnectView.as

```
package com.ui
{
import com.enum.ui.ButtonEnum;
import com.presenter.preload.IPreloadPresenter;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.effects.EffectFactory;
import com.ui.core.effects.GenericMoveEffect;

import flash.display.Bitmap;
import flash.events.MouseEvent;

import org.parade.enum.ViewEnum;

public class ReconnectView extends View
{
private var _loadingImage:Bitmap;
private var _reconnectButton:BitmapButton;

[PostConstruct]
override public function init():void
{
super.init();
presenter.completeSignal.add(onComplete);
_loadingImage = new PreloadView.loadingScreen();
_reconnectButton = UIFactory.getButton(ButtonEnum.GREEN_A, _loadingImage.width, 40,
_loadingImage.x, _loadingImage.y + _loadingImage.height + 5, "RECONNECT");

addListener(_reconnectButton, MouseEvent.CLICK, onReconnect);

addChild(_loadingImage);
addChild(_reconnectButton);

addEffects();
effectsIN();
}

private function onReconnect( e:MouseEvent ):void
{
_reconnectButton.enabled = false;
presenter.transitionToLoad();
}

private function onResize():void
{
_loadingImage.x = 0;
_loadingImage.y
```

```
= 0;

_reconnectButton.x = _loadingImage.x;
_reconnectButton.y = _loadingImage.y + _loadingImage.height + 5;
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.resizeEffect(onResize));
_effects.addEffect(EffectFactory.simpleBackingEffect(1, 0, 0));
_effects.addEffect(EffectFactory.genericMoveEffect(GenericMoveEffect.UP,
GenericMoveEffect.UP, .1, .1));
}

private function onComplete():void { destroy(); }

override public function get height():Number { return 400; }
override public function get width():Number { return 1024; }

[Inject]
public function set presenter( value:IPreloadPresenter ):void { _presenter = value; }
public function get presenter():IPreloadPresenter { return IPreloadPresenter(_presenter); }

override public function get type():String { return ViewEnum.UI; }

override public function destroy():void
{
super.destroy();

_loadingImage.bitmapData.dispose();
_loadingImage = null;
_reconnectButton = UIFactory.destroyButton(_reconnectButton);
}
}
}

-----------------------------------
File 657: igw\com\ui\TransitionView.as
ï»¿package com.ui
{
import com.Application;
import com.enum.PositionEnum;
import com.enum.TimeLogEnum;
import com.event.ServerEvent;
import com.event.StateEvent;
import com.presenter.shared.ITransitionPresenter;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.label.Label;
import com.ui.core.effects.EffectFactory;
import
```

```
com.util.TimeLog;
import com.model.prototype.PrototypeModel;
import com.model.prototype.IPrototype;
import com.util.LoadingScreenHelper;
import com.model.asset.AssetModel;
import com.model.asset.AssetVO;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.display.BitmapData;
import flash.events.TimerEvent;
import flash.filters.GlowFilter;
import flash.utils.Timer;
import flash.events.Event;
import flash.utils.setInterval;
import flash.utils.clearInterval;
import flash.ui.Keyboard;
import flash.display.Stage;
import flash.events.KeyboardEvent;

import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import com.service.language.Localization;

public class TransitionView extends View
{
[Embed(source="LoadScreen.jpg")]
public static var loadingScreen:Class;

[Embed(source="/assets/loading/LS_1.png")]
public static var LoadingScreenBG:Class;

[Embed(source="/assets/loading/loading_main.png")]
public static var LoadingMainBG:Class;

[Embed(source="/assets/loading/loading_percentage.png")]
public static var LoadingPercentageBG:Class;

[Embed(source="/assets/loading/loading_tips_background.png")]
public static var LoadingTipsBG:Class;

[Embed(source="/assets/loading/loading_screen_boarder_right.png")]
public static var BorderRight:Class;

[Embed(source="/assets/loading/loading_screen_boarder_left.png")]
public static var BorderLeft:Class;

private static var LOADING_SCREEN_GROUP_DEFAULT:String = "LoadingScreen_A";
private static var LOADING_SCREEN_ASSET_PATH:String = "assets\\";
private static var LOADING_SCREEN_GROUP:String ="loadingScreenGroup";

private
```

```
var _barSpeedPer100MS:Number = .07;
private var _destroyed:Boolean = false;
private var _label:Label;
private var _loadBar:ProgressBar;
private var _loadingImage:Bitmap;
private var _isEffectComplete:Boolean;
private var _isServerReady:Boolean;
private var _timer:Timer;

private var _loadingImageBG:Bitmap;

private var _loadingMainBG:Bitmap;
private var _loadingPercentageBG:Bitmap;
private var _loadingTipsBG:Bitmap;
private var _tipsLabel:Label;
private var _infoLabel:Label;
private var _titleLabel:Label;
private var _percentageLabel:Label;
private var _borderRight:Bitmap;
private var _borderLeft:Bitmap;

private var _loadingView:Sprite;
private var _borderOverlay:Sprite;

private var _tip:String;
private var _tipKey:String;

private var _titleKey:String;
private var _title:String;

private var _loading:String = 'CodeString.TransitionEvent.Loading'; //Loading
private var _localizationCheckID:int = -1;

private var _stage:Stage;

[Inject]
public var prototypeModel: PrototypeModel;

[Inject]
public var assetModel: AssetModel;

[PostConstruct]
override public function init():void
{

super.init();

if (presenter.failed)
return;

_isEffectComplete
```

```
= false;
_isServerReady = false;
//_loadingImage = new loadingScreen();

presenter.addCompleteListener(onServerReady);
presenter.addUpdateListener(resetEvents);
presenter.removeStateListener(onStateChange);

_loadingView = new Sprite();
_borderOverlay = new Sprite();

//primary loading UI
_getLoadingImageBG();

_loadingTipsBG = new LoadingTipsBG();

_loadingMainBG = new LoadingMainBG();
_loadingMainBG.x = (_loadingImageBG.width / 2) - (_loadingMainBG.width / 2);
_loadingMainBG.y = _loadingImageBG.height - (_loadingMainBG.height +
_loadingTipsBG.height);

_loadingTipsBG.x = (_loadingImageBG.width / 2) - (_loadingTipsBG.width / 2);
_loadingTipsBG.y = _loadingMainBG.y + _loadingMainBG.height;

_loadingPercentageBG = new LoadingPercentageBG();
_loadingPercentageBG.x = _loadingMainBG.x + 504;
_loadingPercentageBG.y = _loadingMainBG.y + 206;

_borderRight = new BorderRight();
_borderRight.x = _loadingImageBG.width - 26;

_borderLeft = new BorderLeft();
_borderLeft.x = 0;

//loading bar
_loadBar = new ProgressBar();
_loadBar.init(ProgressBar.HORIZONTAL, new Bitmap(new BitmapData(1096, 40, false,
0x05ce00)), new Bitmap(new BitmapData(1096, 40, false, 0)), 0);
_loadBar.setMinMax(0, 1);
_loadBar.amount = 0;
_loadBar.base.alpha = 1;
_loadBar.x = _loadingMainBG.x + 116;
_loadBar.y = _loadingMainBG.y + 218;
_loadBar.tweenSpeed = .1;

//text

_infoLabel = new Label(12, 0xFFFFFF, 176, 56, true, 1);
_infoLabel.multiline = false;
_infoLabel.constrictTextToSize = false;
_infoLabel.text
```

```
= _loading;
_infoLabel.y = 6;

_titleLabel = new Label(32, 0xFFFFFF, 600, 56, true, 1);
_titleLabel.multiline = false;
_titleLabel.constrictTextToSize = false;
_titleLabel.x = _loadingMainBG.x + 370;
_titleLabel.y = _loadingMainBG.y + 160;

_tipsLabel = new Label(42, 0xFFFFFF, 1496, 148, true, 1);
_tipsLabel.multiline = true;
_tipsLabel.constrictTextToSize = true;
_tipsLabel.x = _loadingTipsBG.x;
_tipsLabel.y = _loadingTipsBG.y + 8;

_percentageLabel = new Label(22, 0xFFFFFF, 62,28, true, 1);
_percentageLabel.multiline = false;
_percentageLabel.constrictTextToSize = false;
_percentageLabel.text = "0%";
_percentageLabel.x = _loadingPercentageBG.x + 132;
_percentageLabel.y = _loadingPercentageBG.y + 20;

_timer = new Timer(100, 1);
addListener(_timer, TimerEvent.TIMER_COMPLETE, checkLoad);

if (!_destroyed)
{
addEffects();
effectsIN();
} else
{
_timer.start();
}
Application.STAGE.addChild(this);
stage.addEventListener(Event.RESIZE, _onResize);
}

private function _checkForLocalization():void{
if(_localizationCheckID == -1){
_localizationCheckID = setInterval(_checkForLocalization,33);
} else {
_getTip();
_getTitle();
if(_hasTip() && _hasTitle()){
clearInterval(_localizationCheckID);
_updateLabels();
_localizationCheckID = -1;
}
}
}

private
```

```
function _updateLabels():void{
_tipsLabel.text = _tip;
_titleLabel.text = _title;
}

private function _hasTip():Boolean{
return (_tip.length > 0);
}

private function _hasTitle():Boolean{
return (_title.length > 0);
}

private function _getTip():void{
_tip = Localization.instance.getString(_tipKey);
}

private function _getTitle():void{
_title = Localization.instance.getString(_titleKey);
}

private function _getLoadingScreenPrototype():IPrototype{
var prototypes:Vector.<IPrototype> =
_getLoadingScreenPrototypesByGroup(LOADING_SCREEN_GROUP_DEFAULT);

var prototypeSelected:IPrototype;

if(prototypes.length > 0){
prototypeSelected = LoadingScreenHelper.chooseRandomPrototypeByWeight(prototypes);
return prototypeSelected;
}

return null;
}

private function _getLoadingScreenBG(prototype:IPrototype):Bitmap{
var uiAsset:String = prototype.uiAsset;

var asset:AssetVO = assetModel.getAssetVOByName(uiAsset);
_tipKey = asset.descriptionText;
_titleKey = asset.visibleName;
_getTip();
_getTitle();

if(!_hasTip() || !_hasTitle()){
_checkForLocalization();
}
var path:String = LOADING_SCREEN_ASSET_PATH + asset.largeImage;

return LoadingScreenHelper.getBitmap(path);
}
```

```
//todo probably put this is prototype model, we shouldnt do this kind of logic here but, for
//now its going to be ok.
private function _getLoadingScreenPrototypesByGroup(group:String):Vector.<IPrototype>{
var groupPrototypes:Vector.<IPrototype> = new Vector.<IPrototype>;

var allPrototypes = prototypeModel.getLoadingScreenGroupPrototypes();

for(var i:int = 0; i < allPrototypes.length; i++)
{
var prototype:IPrototype = allPrototypes[i];

if(prototype.getValue(LOADING_SCREEN_GROUP) == group){
groupPrototypes.push(prototype);
}
}

return groupPrototypes;
}

private function _getLoadingImageBG():void{
var prototype:IPrototype = _getLoadingScreenPrototype();

if(prototype){
_loadingImageBG = _getLoadingScreenBG(prototype);
} else {
_loadingImageBG = new LoadingScreenBG();
}

}

private function _rescaleWithAspectRatio():void {
var k:Number = stage.stageHeight / stage.stageWidth;
if (_loadingView.width != 0 && _loadingView.width * k > _loadingView.height) {
k = stage.stageWidth / _loadingView.width;
} else {
if (_loadingView.height != 0) {
k = stage.stageHeight / _loadingView.height;
}
}
_loadingView.width *= k;
_loadingView.height *= k;

_borderOverlay.scaleX = _loadingView.scaleX;
_borderOverlay.scaleY = _loadingView.scaleY;
}

private function _centerOnStage():void {
_loadingView.x = (stage.stageWidth / 2) - (_loadingView.width / 2);
_loadingView.y
```

```
= (stage.stageHeight / 2) - (_loadingView.height / 2);

_borderOverlay.x = (stage.stageWidth / 2) - (_borderOverlay.width / 2);
_borderOverlay.y = (stage.stageHeight / 2) - (_borderOverlay.height / 2);

_tipsLabel.x = (_loadingTipsBG.width / 2) - (_tipsLabel.width / 2);

_borderOverlay.visible = _loadingView.width < stage.stageWidth;
}

override public function onEscapePressed():void {}

public function resetEvents():void
{

if (presenter)
{
if (presenter.failed)
destroy();
else if (_isEffectComplete)
{
_isServerReady = Application.CONNECTION_STATE == ServerEvent.AUTHORIZED;
_infoLabel.text = presenter.connectingText;
presenter.sendEvents();
if (_timer.running)
_timer.stop();
if (_isServerReady)
onServerReady();
}
}
}

private function checkLoad( e:TimerEvent ):void
{
var current:Number = _loadBar.amount;
var amount:Number = presenter.estimatedLoadCompleted;
if (amount > current ||
((Application.STATE == StateEvent.GAME_BATTLE || Application.STATE ==
StateEvent.GAME_SECTOR || Application.STATE == StateEvent.GAME_STARBASE) &&
presenter.estimatedLoadCompleted == 0))
_loadBar.amount = current + _barSpeedPer100MS;

if (_destroyed)
destroy()
else if (presenter.hasWaiting)
_timer.start();
else if (_loadBar.amount >= 1)
destroy();
else
_timer.start();

_percentageLabel.text
```

```
= (Math.round(current*100)).toString() + "%";
}

private function onServerReady():void
{
_isServerReady = true;
_infoLabel.text = _loading;
if (_isEffectComplete)
_timer.start();
}

private function _onKeyDown(e:KeyboardEvent):void{
if(e.keyCode == Keyboard.F12){
_infoLabel.visible = !_infoLabel.visible;
}
}

override protected function effectsDoneIn():void
{
_stage = stage;
_infoLabel.visible = false;
stage.addEventListener(KeyboardEvent.KEY_DOWN, _onKeyDown);
_isEffectComplete = true;
if (_isServerReady)
_timer.start();
else
_infoLabel.text = presenter.connectingText;

stage.align = "TL";
stage.scaleMode = "noScale";

addChild(_loadingView);
addChild(_borderOverlay);

_loadingView.addChild(_loadingImageBG);

//addChild(_loadingImage);
_loadingView.addChild(_loadBar);


_updateLabels();

_loadingView.addChild(_loadingMainBG);
_loadingView.addChild(_loadingTipsBG);
_loadingView.addChild(_titleLabel);
_loadingView.addChild(_loadingPercentageBG);
_loadingView.addChild(_percentageLabel);
_loadingView.addChild(_tipsLabel);
_borderOverlay.addChild(_borderLeft);
_borderOverlay.addChild(_borderRight);
addChild(_infoLabel);
```

```
this.alpha = 0;

var fadeTimer:Timer = new Timer(20);
fadeTimer.addEventListener(TimerEvent.TIMER, _fadeIn);
fadeTimer.start()


_onResize(null);

presenter.sendEvents();
}

private function _onDestroy():void
{
super.effectsOUT();

_stage.removeEventListener(Event.RESIZE, _onResize);
_stage.removeEventListener(KeyboardEvent.KEY_DOWN, _onKeyDown);
_stage = null;

var fadeTimer:Timer = new Timer(20);
fadeTimer.addEventListener(TimerEvent.TIMER, _fadeOut);
fadeTimer.start()
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.fullScreenFadeEffect(.3, .3));
//_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.CENTER,
PositionEnum.CENTER, onResize));
//_effects.addEffect(EffectFactory.resizeEffect());
}

private function _onResize(e:Event):void{

_rescaleWithAspectRatio();
_centerOnStage()

_infoLabel.x = stage.stageWidth - _infoLabel.width;

}

/*private function onResize():void
{

x = (DeviceMetrics.WIDTH_PIXELS - (width * scaleX)) * .5;
y = (DeviceMetrics.HEIGHT_PIXELS - (height * scaleX)) * .5;
}*/



private
```

```
function _fadeIn(e:TimerEvent):void{
var timer:Timer = e.currentTarget as Timer;
// increase the alpha value by a small amount each time the timer fires
this.alpha += 0.1;

// stop the animation when the alpha value reaches 1
if (this.alpha >= 1) {
timer.stop();
timer.removeEventListener(TimerEvent.TIMER, _fadeOut);

}
}

private function _fadeOut(e:TimerEvent):void {
var timer:Timer = e.currentTarget as Timer;
// decrease the alpha value by a small amount each time the timer fires
this.alpha -= 0.1;

// stop the animation when the alpha value reaches 0
if (this.alpha <= 0) {
timer.stop();
timer.removeEventListener(TimerEvent.TIMER, _fadeOut);
_destroy();

}
}

[Inject]
public function set presenter( value:ITransitionPresenter ):void { _presenter = value; }
public function get presenter():ITransitionPresenter { return ITransitionPresenter(_presenter); }

override public function get height():Number { return super.height; }
override public function get width():Number { return super.width; }

override public function get typeUnique():Boolean { return true; }
override public function get type():String { return ViewEnum.ALERT; }

private function _destroy():void{

_destroyed = true;
if (presenter == null)
return;
if (Application.CONNECTION_STATE != ServerEvent.NEED_CHARACTER_CREATE)
{
TimeLog.endTimeLog(TimeLogEnum.GAME_LOAD);
TimeLog.enabled = false;
}
presenter.removeUpdateListener(resetEvents);
presenter.removeCompleteListener(onServerReady);

super.destroy();
```

```
_timer.reset();
_timer = null;
_infoLabel.destroy();
_infoLabel = null;
_percentageLabel.destroy();
_percentageLabel = null;
_loadBar.destroy();
_loadBar = null;
_tipsLabel.destroy();
_tipsLabel = null;
_titleLabel.destroy();
_titleLabel = null;


}

override public function destroy():void
{
_onDestroy();

}
}
}

-------------------------------------
File 658: igw\com\ui\UIFactory.as
package com.ui
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.google.analytics.debug.Panel;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.button.ButtonLabelFormatFactory;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.DisplayObject;
import flash.display.Sprite;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

import org.shared.ObjectPool;

public
```

```
class UIFactory
{
private static const _bmdStore:Dictionary = new Dictionary();
private static const _scaleBitmap:ScaleBitmap = new ScaleBitmap();
private static const _scaleRect:Rectangle = new Rectangle();


//==========================================================================
//**************************************************************************************************
// PANELS
//**************************************************************************************************

//==========================================================================

/**
* Essentially the same as <code>getScaleBitmap</code> but also resizes and positions the
ScaleBitmap
*
* @param type The type of panel to create. All panels are defined in PanelEnum
* @param width Width to resize the panel to
* @param height Height to resize the panel to
* @param x X position of the panel
* @param y Y position of the panel
* @return Returns a newly created Panel (ScaleBitmap)
*/
public static function getPanel( type:String,
width:Number = 0, height:Number = 0,
x:int = 0, y:int = 0 ):ScaleBitmap
{
var bitmap:ScaleBitmap = getScaleBitmap(type);
bitmap.setSize(width, height);
bitmap.x = x;
bitmap.y = y;

return bitmap;
}

/**
* Creates a panel with a header and optional title label.
*
* @param panelType The type of panel to create. All panels are defined in PanelEnum
* @param headerType The type of header to create. All panels are defined in PanelEnum
* @param width Width to resize the panel and header to
* @param height Height to resize the panel to
* @param headerHeight Height to resize the header to
* @param x X position of the header panel
* @param y Y position of the header panel
* @param text The text to display in the title of the header
* @param labelType The format to apply to the label
* @return Returns a newly creaded Header Panel (Sprite)
*/
```

```
public static function getHeaderPanel( panelType:String, headerType:String,
width:Number = 0, height:Number = 0, headerHeight:Number = 0,
x:int = 0, y:int = 0,
text:String = null, labelType:String = null ):Sprite
{
var sprite:Sprite = new Sprite();

//create the back panel
var panel:ScaleBitmap = getPanel(panelType, width, height, 0, headerHeight);
sprite.addChild(panel);

//adjust based on panel
switch (panelType)
{
case PanelEnum.CONTAINER_DOUBLE_NOTCHED_ARROWS:
case PanelEnum.CONTAINER_RIGHT_NOTCHED_ARROW:
width -= 2;
x += 1;
break;
case PanelEnum.CONTAINER_INNER:
panel.y -= 2;
break;
}

//create the header
var header:ScaleBitmap = getPanel(headerType, width, headerHeight, 0, 0);
if (headerType == PanelEnum.HEADER_NOTCHED_RIGHT)
{
header.width += 1;
header.scaleX = -1;
header.x = header.width;
}
sprite.addChild(header);

//add the text
if (text)
{
if (labelType == null)
labelType = LabelEnum.H2;
var label:Label = getLabel(labelType, width, headerHeight, 4, 4);
label.align = TextFormatAlign.LEFT;
label.text = text;
label.x = 10;
label.y = (headerHeight - label.textHeight) * .5;
sprite.addChild(label);
}
sprite.x = x;
sprite.y = y;

return
```

```
sprite;
}

public static function destroyPanel( panel:* ):*
{
if (!panel)
return null;
if (panel is Sprite)
{
var sprite:Sprite = Sprite(panel);
while (sprite.numChildren > 0)
sprite.removeChildAt(0);
} else
{
var bitmap:Bitmap = Bitmap(panel);
bitmap.bitmapData = null;
}
ObjectPool.give(panel);
return null;
}


//==========================================================================
//**************************************************************************************************
// LABELS
//**************************************************************************************************

//==========================================================================

/**
* Creates a new Label
*
* @param type The type of label to create. All labels are defined in LabelEnum
* @param width Width to resize the label to
* @param height Height to resize the label to
* @param x X position of the label
* @param y Y position of the label
* @return Returns a newly created Label
*/
public static function getLabel( type:String, width:Number, height:Number, x:int = 0, y:int = 0
):Label
{
var label:Label = ObjectPool.get(Label);

switch (type)
{
case LabelEnum.H1:
label.init(30, 0xd1e5f7, width, height);
label.bold = true;
break;

case
```

```
LabelEnum.H2:
label.init(26, 0xd1e5f7, width, height);
label.bold = true;
break;

case LabelEnum.H3:
label.init(22, 0xd1e5f7, width, height);
label.bold = true;
break;
case LabelEnum.H4:
label.init(18, 0xacd1ff, width, height);
label.bold = true;
break;
case LabelEnum.H5:
label.init(16, 0xacd1ff, width, height);
label.bold = true;
break;

case LabelEnum.SUBTITLE:
label.init(20, 0xd1e5f7, width, height);
break;

case LabelEnum.TITLE:
label.init(28, 0xd1e5f7, width, height);
break;

case LabelEnum.DESCRIPTION:
label.init(12, 0xecffff, width, height, true, 1);
label.bold = false;
label.multiline = true;
label.align = TextFormatAlign.LEFT;
label.constrictTextToSize = true;
break;

case LabelEnum.DEFAULT_OPEN_SANS:
label.init(14, 0xf0f0f0, width, height, true, 1);
break;

case LabelEnum.DEFAULT:
default:
label.init(14, 0xefefef, width, height);
break;
}

label.x = x;
label.y = y;
label.constrictTextToSize = false;
return label;
}

public
```

```
static function destroyLabel( label:Label ):*
{
if (!label)
return null;
ObjectPool.give(label);
return null;
}


//==========================================================================================
//******************************************************************************************
// BUTTON
//******************************************************************************************

//==========================================================================================

/**
* Creates a new button
* @param type
* @param width
* @param height
* @param x
* @param y
* @param text
* @param labelType
* @return
*
*/
public static function getButton( type:String,
width:Number = 0, height:Number = 0,
x:int = 0, y:int = 0,
text:String = null, labelType:String = null ):BitmapButton
{
var button:BitmapButton = ObjectPool.get(BitmapButton);
initButton(button, type);
button.labelType = labelType;
if (setScaleRect(type))
button.scale9Grid = _scaleRect;

switch (type)
{
case ButtonEnum.BLUE_A:
case ButtonEnum.GOLD_A:
case ButtonEnum.RED_A:
case ButtonEnum.GREEN_A:
if (!labelType)
button.labelType = LabelEnum.H2;
button.labelFormat = ButtonLabelFormatFactory.getFormat(type);
break;
case ButtonEnum.ACCORDIAN_SUBITEM:
case
```

```
ButtonEnum.HEADER:
case ButtonEnum.HEADER_NOTCHED:
if (!labelType)
button.labelType = LabelEnum.H2;
button.labelFormat = ButtonLabelFormatFactory.getFormat(type);
break;
case ButtonEnum.CHECKBOX:
button.selectable = true;
break;
}

button.setSize(width, height);
button.x = x;
button.y = y;
if (text)
button.text = text;

return button;
}

private static function initButton( button:BitmapButton, type:String ):void
{
var up:BitmapData = getBitmapData(type + "UpBMD");
var down:BitmapData = getBitmapData(type + "DownBMD");
if (!down)
down = up;
var ro:BitmapData = getBitmapData(type + "ROBMD");
if (!ro)
ro = (down) ? down : up;
var disabled:BitmapData = getBitmapData(type + "DisabledBMD");
if (!disabled)
disabled = (down) ? down : up;
var selected:BitmapData = getBitmapData(type + "SelectedBMD");
if (!selected)
selected = (down) ? down : (disabled) ? disabled : (ro) ? ro : up;
button.init(up, ro, down, disabled, selected);
}

public static function destroyButton( button:BitmapButton ):*
{
if (button)
ObjectPool.give(button);
return null;
}


//=============================================================================
//*****************************************************************************
// PROGRESS BAR
//*****************************************************************************

//=============================================================================
```

```
/**
* Creates a new progress bar
*
* @param barBmpd the BitmapdData that will fill as the bar progresses
* @param bkgdBmpd the BitmapData that is used as the background for the progress bar
* @param min the minimum value for the progress bar
* @param max the maximum value for the progress bar
* @param amount the current value of the progress bar
* @return ProgressBar
*/
public static function getProgressBar( bar:DisplayObject, background:DisplayObject,
min:Number = 0, max:Number = 1, amount:Number = 0, x:Number = 0, y:Number = 0
):ProgressBar
{
var pbar:ProgressBar = ObjectPool.get(ProgressBar);
pbar.init(ProgressBar.HORIZONTAL, bar, background);
pbar.setMinMax(min, max);
pbar.amount = amount;
pbar.x = x;
pbar.y = y;

return pbar;
}

public static function destroyProgressBar( bar:ProgressBar ):*
{
ObjectPool.give(bar);
return null;
}


//==========================================================================
//**************************************************************************************************
// GENERAL
//**************************************************************************************************

//==========================================================================

public static function getBitmap( name:String ):Bitmap
{
var bitmap:Bitmap = new Bitmap();
if (!_bmdStore.hasOwnProperty(name))
getBitmapData(name);
bitmap.bitmapData = _bmdStore[name];
return bitmap;
}

public static function getScaleBitmap( name:String ):ScaleBitmap
{
```

```
var bitmap:ScaleBitmap = new ScaleBitmap();
if (!_bmdStore.hasOwnProperty(name))
getBitmapData(name);
bitmap.bitmapData = _bmdStore[name];
//see if there is a scale rect for this type and if so, apply it
if (setScaleRect(name))
bitmap.scale9Grid = _scaleRect;
return bitmap;
}

public static function getBitmapData( name:String ):BitmapData
{
if (_bmdStore.hasOwnProperty(name))
return _bmdStore[name];
try
{
var bmdClass:Class = Class(getDefinitionByName(name));
} catch ( e:Error )
{
return null;
}
var bmd:BitmapData = BitmapData(new bmdClass());
_bmdStore[name] = bmd;
return bmd;
}

public static function getDefaultWindow():void
{

}

private static function setScaleRect( type:String ):Boolean
{
switch (type)
{
case ButtonEnum.FRAME_BLUE:
case ButtonEnum.FRAME_GREEN:
case ButtonEnum.FRAME_GREY:
case ButtonEnum.FRAME_RED:
case ButtonEnum.FRAME_GOLD:
case ButtonEnum.BLUE_A:
case ButtonEnum.GOLD_A:
case ButtonEnum.RED_A:
case ButtonEnum.GREEN_A:
case ButtonEnum.GREY:
_scaleRect.setTo(10, 10, 2, 2);
return true;
case ButtonEnum.CHARACTER_FRAME:
case PanelEnum.CHARACTER_FRAME:
case
```

```
PanelEnum.BLUE_FRAME:
case ButtonEnum.ICON_FRAME:
_scaleRect.setTo(12, 12, 8, 8);
return true;
case ButtonEnum.DROP_TAB:
_scaleRect.setTo(13, 1, 4, 4);
return true;
case PanelEnum.HEADER:
case PanelEnum.HEADER_NOTCHED:
case PanelEnum.NUMBER_BOX:
case ButtonEnum.HEADER:
case ButtonEnum.HEADER_NOTCHED:
_scaleRect.setTo(9, 9, 4, 4);
return true;
case PanelEnum.STATBAR_CONTAINER:
case PanelEnum.STATBAR:
case PanelEnum.STATBAR_GREY:
case PanelEnum.SCROLL_BAR:
case PanelEnum.CONTAINER_INNER:
case PanelEnum.CONTAINER_INNER_DARK:
_scaleRect.setTo(1, 1, 1, 1);
return true;
case PanelEnum.CONTAINER_DOUBLE_NOTCHED:
case PanelEnum.CONTAINER_RIGHT_NOTCHED_ARROW:
case PanelEnum.CONTAINER_DOUBLE_NOTCHED_ARROWS:
case PanelEnum.CONTAINER_NOTCHED:
case PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL:
case PanelEnum.CONTAINER_NOTCHED_LEFT_SMALL:
_scaleRect.setTo(30, 2, 2, 2);
return true;
case PanelEnum.PLAYER_CONTAINER_NOTCHED:
_scaleRect.setTo(20, 7, 2, 2);
return true;
case PanelEnum.ENEMY_CONTAINER_NOTCHED:
_scaleRect.setTo(11, 2, 2, 2);
return true;
case PanelEnum.WINDOW:
_scaleRect.setTo(50, 125, 4, 4);
return true;
case PanelEnum.WINDOW_HEADER:
_scaleRect.setTo(30, 30, 4, 4);
return true;
case PanelEnum.WINDOW_SIDE_HEADER:
_scaleRect.setTo(18, 130, 2, 2);
return true;
case PanelEnum.INPUT_BOX_BLUE:
case PanelEnum.INPUT_BOX_GOLD:
case PanelEnum.BOX_GOLD_GLOW:
case PanelEnum.LEADERBOARD_ROW_GLOW:
_scaleRect.setTo(5, 5, 5, 5);
return
```

```
true;
case PanelEnum.FAQ_SUBJECT_BG:
_scaleRect.setTo(0, 7, 522, 29);
return true;
default:
_scaleRect.setTo(0, 0, 1, 1);
return true;
}
return false;
}
}
}

-----------------------------------
File 659: igw\com\ui\alert\AlertView.as
package com.ui.alert
{
import com.controller.keyboard.KeyboardKey;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.parade.enum.ViewEnum;

public class AlertView extends View
{
protected var _bg:Sprite;
protected var _closeBtn:BitmapButton;
protected var _closeCallback:Function;

protected var _viewName:Label;
protected var _bodyText:Label;

protected var _btnOne:BitmapButton;
protected var _btnOneArgs:Array;
protected var _btnOneCallback:Function;

protected var _btnTwo:BitmapButton;
protected var _btnTwoArgs:Array;
protected var _btnTwoCallback:Function;

protected var _MAX_WIDTH:int = 300;

[PostConstruct]
override
```

```
public function init():void
{
super.init();

_closeBtn = ButtonFactory.getCloseButton(0, 0);
_closeBtn.scaleX = _closeBtn.scaleY = .75;
addListener(_closeBtn, MouseEvent.CLICK, onClose);

_keyboard.addKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);

layout();

addEffects();
effectsIN();
}

public function setUp( args:Array ):void
{
var btnOneText:String = args[2];
var btnTwoText:String = args[5];
_btnOneArgs = new Array();
_btnTwoArgs = new Array();

var windowBG:Class = Class(getDefinitionByName(('WindowContextMenuMC')));
_bg = Sprite(new windowBG());
addChild(_bg);

_viewName = new Label(18, 0xffffff, _bg.width - 60, 30);
_viewName.align = TextFormatAlign.LEFT;
_viewName.text = args[0];
addChild(_viewName);

_bodyText = new Label(18, 0xffffff);
_bodyText.x = 25;
_bodyText.align = TextFormatAlign.CENTER;
_bodyText.constrictTextToSize = false;
_bodyText.multiline = true;
_bodyText.htmlText = args[1];
_bodyText.setSize(_bg.width - 60, _bodyText.textHeight);
addChild(_bodyText);

if (btnOneText != '' && btnOneText != null)
{
_btnOne = ButtonFactory.getBitmapButton('LeftBtnUpBMD', 0, 0, btnOneText, 0xFFFFFF,
'LeftBtnRollOverBMD', 'LeftBtnDownBMD', 'LeftBtnDownBMD', null, 10);
_btnOne.addEventListener(MouseEvent.CLICK, onBtnOneClick, false, 0, true);
_btnOneCallback = args[3];
_btnOneArgs = args[4];
addChild(_btnOne);
}

if
```

```
(btnTwoText != '' && btnTwoText != null)
{
_btnTwo = ButtonFactory.getBitmapButton('RightBtnUpBMD', 0, 0, btnTwoText, 0xFFFFFF,
'RightBtnRollOverBMD', 'RightBtnDownBMD', 'RightBtnDownBMD', null, 10);
_btnTwo.addEventListener(MouseEvent.CLICK, onBtnTwoClick, false, 0, true);
_btnTwoCallback = args[6];
_btnTwoArgs = args[7];
addChild(_btnTwo);
}

if (args[8])
_closeCallback = btnTwoPress;
else
_closeCallback = btnOnePress;
}

protected function onEnterPress( keyCode:uint ):void
{
}

protected function layout():void
{
_viewName.x = 28;
_viewName.y = 14;

_closeBtn.x = _bg.width - 33;
_closeBtn.y = _viewName.y + 5;

_bodyText.y = _viewName.y + _viewName.height + 20;

var height:Number;
if (_btnOne != null)
{
_btnOne.x = 30;
_btnOne.y = _bg.height - 60;
height = _btnOne.y + _btnOne.height + 5
} else
height = _bodyText.y + _bodyText.height + 5

if (_btnTwo != null)
{
_btnTwo.x = _bg.width - (_btnTwo.width + 20);
_btnTwo.y = _bg.height - 60;
} else if (_btnOne)
_btnOne.x = (_bg.width - _btnOne.width) / 2;
addChild(_closeBtn);
}

protected function onBtnOneClick( e:MouseEvent ):void
{
btnOnePress();
```

```
destroy();
}

protected function btnOnePress():void
{
if (_btnOneCallback != null)
{
if (_btnOneArgs != null)
_btnOneCallback(_btnOneArgs);
else
_btnOneCallback();
}
}

protected function onBtnTwoClick( e:MouseEvent ):void
{
btnTwoPress();
destroy();
}

protected function btnTwoPress():void
{
if (_btnTwoCallback != null)
{
if (_btnTwoArgs != null)
_btnTwoCallback(_btnTwoArgs);
else
_btnTwoCallback();
}
}

override protected function onClose( e:MouseEvent = null ):void
{
if (_closeCallback != null)
_closeCallback();
super.onClose(e);
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function get type():String { return ViewEnum.ALERT; }
override public function get typeUnique():Boolean { return true; }

override public function destroy():void
{

_bg = null;
_closeBtn.destroy();
_closeBtn
```

```
= null;
_closeCallback = null;

if (_btnOne)
{
_btnOne.removeEventListener(MouseEvent.CLICK, onBtnOneClick);
_btnOne.destroy();
_btnOne = null;
}

if (_btnTwo)
{
_btnTwo.removeEventListener(MouseEvent.CLICK, onBtnTwoClick);
_btnTwo.destroy();
_btnTwo = null;
}

_viewName.destroy();
_viewName = null;

_bodyText.destroy();
_bodyText = null;

_keyboard.removeKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);
super.destroy()
}
}
}

-----------------------------------
File 660: igw\com\ui\alert\AttackAlert.as
package com.ui.alert
{
import com.model.fleet.FleetVO;
import com.presenter.starbase.IAttackAlertPresenter;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class AttackAlert extends View
{
```

```
public var battleServerAddress:String;
public var fleetID:String;
public var fleetName:String;

private var _bg:Bitmap;
private var _bodyText:Label;
private var _closeButton:BitmapButton;
private var _defendButton:BitmapButton;
private var _ignoreButton:BitmapButton;
private var _margin:int = 30;
private var _title:Label;

private var _fleetBattleTitleText:String = 'CodeString.Alert.Battle.Title'; //INCOMING ATTACK
private var _fleetBattleBodyText:String = 'CodeString.Alert.FleetBattle.Body'; //Your fleet is
under attack! Would you like to defend it?
private var _baseBattleTitleText:String = 'CodeString.Alert.Battle.Title'; //INCOMING ATTACK
private var _baseBattleBodyText:String = 'CodeString.Alert.BaseBattle.Body'; //Your fleet is
under attack! Would you like to defend it?
private var _viewBtnText:String = 'CodeString.Alert.Battle.ViewBtn'; //Defend
private var _dontViewBtnText:String = 'CodeString.Alert.Battle.DontViewBtn'; //Ignore

[PostConstruct]
public override function init():void
{
super.init();
presenter.addFleetUpdateListener(onUpdate);

_bg = PanelFactory.getPanel("WindowContextMenuBMD");

_title = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DIALOG_TITLE, _bg.width -
_margin);
_title.align = TextFormatAlign.LEFT;
_title.text = (fleetID != null) ? _fleetBattleTitleText : _baseBattleTitleText;
_title.x = _margin;
_title.y = 13;

_bodyText = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, _bg.width - 2 *
_margin - 10, 18);
_bodyText.align = TextFormatAlign.CENTER;
_bodyText.constrictTextToSize = false;
_bodyText.multiline = true;
_bodyText.htmlText = (fleetID != null) ? _fleetBattleBodyText : _baseBattleBodyText;
_bodyText.setSize(_bodyText.width, _bodyText.textHeight);
_bodyText.x = _margin;
_bodyText.y = _title.x + _title.height + 15;

_closeButton = ButtonFactory.getCloseButton(_bg.width - 40, 15);

_defendButton = ButtonFactory.getBitmapButton('LeftBtnUpBMD', x, y, _viewBtnText,
0xFFFFFF,
```

```
'LeftBtnRollOverBMD', 'LeftBtnDownBMD', '', '', 12, 0);
_defendButton.x = (_bg.width * .5) - _defendButton.width - _margin;
_defendButton.y = _bg.height - _defendButton.height - _margin;

_ignoreButton = ButtonFactory.getBitmapButton('RightBtnUpBMD', x, y, _dontViewBtnText,
0xFFFFFF, 'RightBtnRollOverBMD', 'RightBtnDownBMD', '', '', 12, 0);
_ignoreButton.x = (_bg.width * .5) + _margin;
_ignoreButton.y = _bg.height - _ignoreButton.height - _margin;

addListener(_closeButton, MouseEvent.CLICK, onIgnore);
addListener(_defendButton, MouseEvent.CLICK, onDefend);
addListener(_ignoreButton, MouseEvent.CLICK, onIgnore);

addChild(_bg);
addChild(_title);
addChild(_bodyText);
addChild(_closeButton);
addChild(_defendButton);
addChild(_ignoreButton);

addEffects();
effectsIN();
}

private function onDefend( e:MouseEvent ):void
{
presenter.removeAllAlerts(AttackAlert);
presenter.joinBattle(battleServerAddress, fleetID);
destroy();
}

private function onIgnore( e:MouseEvent ):void
{
destroy();
}

private function onUpdate( fleet:FleetVO = null ):void
{
if (presenter.hasBattleEnded(battleServerAddress, fleetID))
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IAttackAlertPresenter ):void { _presenter = value; }
public function get presenter():IAttackAlertPresenter { return IAttackAlertPresenter(_presenter); }

override public function get type():String { return ViewEnum.ALERT; }
override
```

```
public function get typeUnique():Boolean { return true; }

override public function destroy():void
{
presenter.removeFleetUpdateListener(onUpdate);
super.destroy();

_bg = null;
_bodyText.destroy();
_bodyText = null;
ObjectPool.give(_closeButton);
_closeButton = null;
ObjectPool.give(_defendButton);
_defendButton = null;
ObjectPool.give(_ignoreButton);
_ignoreButton = null;
_title.destroy();
_title = null;
}
}
}

-------------------------------------
File 661: igw\com\ui\alert\ConfirmationView.as
package com.ui.alert
{
import com.enum.ui.LabelEnum;
import com.ui.UIFactory;
import com.ui.core.ButtonPrototype;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.events.MouseEvent;

import org.shared.ObjectPool;

public class ConfirmationView extends View
{
private const DIALOG_MARGIN:int = 20;
private const DIALOG_LEFT_MARGIN:int = 30;
private const DIALOG_RIGHT_MARGIN:int = 19;
private const DIALOG_BOTTOM_MARGIN:int = 50;

// Button sizes
protected const SIZE_LARGE:String = 'Large';
protected const SIZE_MEDIUM:String = 'Medium';
protected const SIZE_SMALL:String = 'Small';

private
```

```
var _bg:DefaultWindowBG;
private var _buttons:Vector.<BitmapButton>;
private var _bodyText:Label;
private var _buttonProtos:Vector.<ButtonPrototype>;

[PostConstruct]
public override function init():void
{
super.init();

//build and layout the buttons
_buttons = new Vector.<BitmapButton>;
var button:BitmapButton;
for (var i:int = 0; i < _buttonProtos.length; i++)
{
button = UIFactory.getButton(_buttonProtos[i].type, 180, 40, 0, _bg.height + 8,
_buttonProtos[i].text);
if (i == 0)
button.x = _bg.width - button.width - 8;
else
button.x = _buttons[i - 1].x - button.width - 8;
_buttons.push(button);
addChild(button);
addListener(button, MouseEvent.CLICK, onButtonClicked);
}

addEffects();
effectsIN();
}

public function setup( title:String, body:String, buttons:Vector.<ButtonPrototype> ):void
{
_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(500, 230);
_bg.addTitle(title, 145);

_bodyText = UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, _bg.width - 60,
_bg.height, 30, 120);
_bodyText.fontSize = 20;
_bodyText.multiline = true;
_bodyText.htmlText = body;

_buttonProtos = buttons;

addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

addChild(_bg);
addChild(_bodyText);
}

protected
```

```
function onButtonClicked( e:MouseEvent ):int
{
var idx:int = _buttons.indexOf(BitmapButton(e.currentTarget));
var proto:ButtonPrototype = _buttonProtos[idx];

if (proto.callback != null)
proto.callback.apply(null, proto.args);

if (proto.doClose)
destroy();

return idx;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function destroy():void
{
super.destroy();
}
}
}
```

File 662: igw\com\ui\alert\DropBubbleView.as

```
package com.ui.alert
{
import com.presenter.sector.ISectorPresenter;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.ash.core.Entity;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class DropBubbleView extends View
{
public var entity:Entity;

private var _bg:Bitmap;
private var _body:Label;
private
```

```
var _closeButton:BitmapButton;
private var _retreatButton:BitmapButton;
private var _attackButton:BitmapButton;
private var _margin:int = 30;
private var _title:Label;

private var _titleText:String = 'CodeString.BubbleDropView.Title'; //PROTECTION ALERT
private var _bodyText:String = 'CodeString.BubbleDropView.Body'; //You will lose your base
protection if you attack this base. <br\><br\>Are you sure you wish to continue?
private var _retreatText:String = 'CodeString.BubbleDropView.Retreat'; //Retreat
private var _attackText:String = 'CodeString.BubbleDropView.Attack'; //Attack

[PostConstruct]
public override function init():void
{
super.init();
_bg = PanelFactory.getPanel("WindowContextMenuBMD");

_title = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DIALOG_TITLE, _bg.width -
_margin);
_title.align = TextFormatAlign.LEFT;
_title.text = _titleText;
_title.x = _margin;
_title.y = 13;

_body = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, _bg.width - 2 *
_margin - 10, 18);
_body.align = TextFormatAlign.CENTER;
_body.constrictTextToSize = false;
_body.leading = -4;
_body.multiline = true;
_body.htmlText = _bodyText;
_body.setSize(_body.width, _body.textHeight);
_body.x = _margin;
_body.y = 65;

_closeButton = ButtonFactory.getCloseButton(_bg.width - 40, 15);

_retreatButton = ButtonFactory.getBitmapButton('LeftBtnUpBMD', x, y, _retreatText,
0xFFFFFFF, 'LeftBtnRollOverBMD', 'LeftBtnDownBMD', '', '', 12, 0);
_retreatButton.x = (_bg.width * .5) - _retreatButton.width - _margin;
_retreatButton.y = _bg.height - _retreatButton.height - _margin;

_attackButton = ButtonFactory.getBitmapButton('RightBtnUpBMD', x, y, _attackText,
0xFFFFFFF, 'RightBtnRollOverBMD', 'RightBtnDownBMD', '', '', 12, 0);
_attackButton.x = (_bg.width * .5) + _margin;
_attackButton.y = _retreatButton.y;

addListener(_closeButton, MouseEvent.CLICK, onClose);
addListener(_retreatButton, MouseEvent.CLICK, onClose);
addListener(_attackButton,
```

```
MouseEvent.CLICK, onAttack);

addChild(_bg);
addChild(_title);
addChild(_body);
addChild(_closeButton);
addChild(_retreatButton);
addChild(_attackButton);

addEffects();
effectsIN();
}

private function onAttack( e:MouseEvent ):void
{
presenter.attackEntity(entity, true);
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:ISectorPresenter ):void { _presenter = value; }
public function get presenter():ISectorPresenter { return ISectorPresenter(_presenter); }

override public function get type():String { return ViewEnum.ALERT; }
override public function get typeUnique():Boolean { return true; }

override public function destroy():void
{
super.destroy();

entity = null;
_bg = null;
_body.destroy();
_body = null;
ObjectPool.give(_closeButton);
_closeButton = null;
ObjectPool.give(_retreatButton);
_retreatButton = null;
ObjectPool.give(_attackButton);
_attackButton = null;
_title.destroy();
_title = null;
}
}
}
```

File

663: igw\com\ui\alert\FTEOverlayView.as

```
package com.ui.alert
{
import com.Application;
import com.controller.fte.FTEStepVO;
import com.enum.PositionEnum;
import com.presenter.shared.IUIPresenter;
import com.ui.core.View;
import com.ui.core.effects.EffectFactory;
import com.ui.core.effects.ViewEffects;
import com.ui.modal.mission.FTEDialogueView;
import com.ui.modal.store.StoreTransactionPage;

import flash.display.DisplayObject;
import flash.display.DisplayObjectContainer;
import flash.display.MovieClip;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.geom.Point;
import flash.geom.Rectangle;
import flash.utils.Timer;
import flash.utils.describeType;
import flash.utils.getDefinitionByName;

import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;

public class FTEOverlayView extends View
{
private var _arrow:MovieClip;
private var _clickToContinue:Boolean = false;
private var _dialogueView:FTEDialogueView;
private var _fteStepVO:FTEStepVO;
private var _ignoreStoreOffset:Boolean = false;
private var _originalApplicationSize:Point;
private var _overlay:Sprite;
private var _rolloverListener:Boolean = false;
private var _trigger:DisplayObject;
private var _triggerTimer:Timer;
private var _view:*;
private var _viewAlreadyExisted:Boolean = false;

[PostConstruct]
override public function init():void
{
super.init();

_mute = true;
_overlay
```

```
= new Sprite();
addChild(_overlay);

var arrowClass:Class = Class(getDefinitionByName('ArrowMC'));
_arrow = MovieClip(new arrowClass());
_arrow.visible = false;
addChild(_arrow);

_originalApplicationSize = new Point(DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS);
_triggerTimer = new Timer(500, 1);
_triggerTimer.addEventListener(TimerEvent.TIMER_COMPLETE, showCutout, false, 0, true);

//see if the view has any effects. we want to listen to the done event of the effect to reposition
the cutout
if (_view && _view.effects != null)
{
ViewEffects(_view.effects).addInListener(onEffectDoneIn);
}

if (_view && DisplayObject(_view).visible == false)
DisplayObject(_view).visible = true;

if (!_view || !_view.effects || _view.effects.numEffects == 0 || _view.effects.isDoneIn ||
_viewAlreadyExisted)
{
//go ahead and show the cutout since we dont have to wait for the view to animate in or the view
was already done
onEffectDoneIn();
if (!_view && (_fteStepVO.cutout != null || _fteStepVO.arrowPosition != null))
{
Application.STAGE.removeEventListener(Event.RESIZE, onResize);
Application.STAGE.addEventListener(Event.RESIZE, onResize, false, 0, true);
}
}

removeListener(Application.STAGE, MouseEvent.CLICK, onTriggerClicked);
if (_clickToContinue)
addListener(Application.STAGE, MouseEvent.CLICK, onTriggerClicked, null, false, 2);

effectsIN();
}

/**
* Called when the target view is done animating in
*/
protected function onEffectDoneIn():void
{
showCutout();
showArrow();
}
```

```
private function onTriggerClicked( e:MouseEvent ):void
{
if (_clickToContinue && _dialogueView)
{
presenter.fteNextStep();
} else if ((_clickToContinue && !e.currentTarget is FTEOverlayView) || e.currentTarget ==
_trigger)
{
presenter.fteNextStep();
}
}

/**
* Recursively crawls through the view to try and determine what the trigger is supposed to be
* @param cutout The region of the overlay that the user can click through
* @param container A display object that has one or more children
* @return The target trigger
*/
private function findTrigger( cutout:Rectangle, container:DisplayObjectContainer ):DisplayObject
{
var displayObject:DisplayObject;
var intersection:Rectangle;
var rect:Rectangle;
var newRect:Rectangle;
var testObj:DisplayObject;
for (var i:int = 0; i < container.numChildren; i++)
{
displayObject = container.getChildAt(i);

if (!displayObject.visible)
continue;

rect = displayObject.getRect(_view);
intersection = rect.intersection(cutout);
var diff:Number = (intersection.width / cutout.width) * (intersection.height / cutout.height);
var xml:XML = describeType(displayObject);
if (String(xml.@name) == "com.ui.modal.store::StoreTransactionPage")
{
var page:StoreTransactionPage = StoreTransactionPage(displayObject);
if (page.visibleItemsContainer && page.visibleItemsContainer.scrollRect &&
!_ignoreStoreOffset)
return findTrigger(cutout, page.visibleItemsContainer);
}

if (cutout.intersects(rect) && diff > .9)
{
switch (String(xml.@name))
{
case
```

```
"com.ui.core.component.page::PageComponent":
case "com.ui.core.component.page::Page":
case "flash.display::Sprite":
case "com.ui.hud.shared.bridge.right::TransactionRiver":
case "com.ui.modal.store::StoreTransactionPage":
testObj = findTrigger(cutout, DisplayObjectContainer(displayObject));
if (testObj)
return testObj;
break;
case "com.ui.core.component.page::PageIcon":
case "com.ui.core.component.contextmenu::ContextMenuComponent":
case "com.ui.modal.dock::ShipButton":
case "com.ui.modal.store::StoreTransactionButton":
case "com.ui.modal.store::StoreItem":
case "com.ui.hud.shared.bridge.right::TransactionButton":
case "com.ui.modal.construction::ConstructionItem":
return displayObject;
case "com.ui.core.component.button::BitmapButton":
testObj = findTrigger(cutout, DisplayObjectContainer(displayObject));
if (testObj)
return testObj;
else
return displayObject;
break;
case "flash.display::Bitmap":
break;
}
switch (String(xml.@base))
{
case "com.ui.modal.store::StorePage":
newRect = cutout.clone();
newRect.left += 300;
testObj = findTrigger(newRect, DisplayObjectContainer(displayObject));
if (testObj)
return testObj;
break;

case "com.ui.core.component.page::PageComponent":
case "com.ui.core.component.page::Page":
case "flash.display::Sprite":
case "com.ui.modal.store::StorePage":
testObj = findTrigger(cutout, DisplayObjectContainer(displayObject));
if (testObj)
return testObj;
break;
// return StorePage(displayObject).itemsComponents[0].buyBtn;
case "com.ui.core.component.page::PageIcon":
case "com.ui.core.component.contextmenu::ContextMenuComponent":
return displayObject;
case "com.ui.core.component.button::BitmapButton":
testObj
```

```
= findTrigger(cutout, DisplayObjectContainer(displayObject));
if (testObj)
return testObj;
else
return displayObject;
break;
case "flash.display::DisplayObject":
break;
}
}
}
return null;
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.TOP,
onResize));
}

/**
* Called when the stage is resized to hide the cutout until the target view is done repositioning
*/
protected function onResize( e:Event = null ):void
{
_overlay.graphics.clear();
_overlay.graphics.beginFill(0xffffff, 0.0);
_overlay.graphics.drawRect(0, 0, DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS);
if (!_view)
{
var xdiff:Number = (_originalApplicationSize.x - DeviceMetrics.WIDTH_PIXELS) * .5;
var ydiff:Number = (_originalApplicationSize.y - DeviceMetrics.HEIGHT_PIXELS) * .5;
if (_fteStepVO.cutout)
{
_fteStepVO.cutout.x = _fteStepVO.cutout.x - xdiff;
_fteStepVO.cutout.y = _fteStepVO.cutout.y - ydiff;
_overlay.graphics.drawRect(_fteStepVO.cutout.x, _fteStepVO.cutout.y,
_fteStepVO.cutout.width, _fteStepVO.cutout.height);
}
if (_fteStepVO.arrowPosition)
{
_fteStepVO.arrowPosition.x = _fteStepVO.arrowPosition.x - xdiff;
_fteStepVO.arrowPosition.y = _fteStepVO.arrowPosition.y - ydiff;
_arrow.x = _fteStepVO.arrowPosition.x;
_arrow.y = _fteStepVO.arrowPosition.y;
}
_originalApplicationSize.setTo(DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS);
}
_overlay.graphics.endFill();
```

```
}

public function showCutout( e:TimerEvent = null ):void
{
//remove the listener from the current trigger
if (_trigger)
{
//remove the glow filter
if (_trigger.filters.length > 0)
_trigger.filters = [];
removeListener(_trigger, MouseEvent.CLICK, onTriggerClicked);
}

_overlay.graphics.clear();
_overlay.graphics.beginFill(0xffffff, 0.0);
_overlay.graphics.drawRect(0, 0, DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS);
if (_fteStepVO.cutout)
{
if (_view)
{
// Enable for cutout debugging
// _overlay.graphics.lineStyle(2, Color.rgb(0, 255, 0));
_trigger = findTrigger(_fteStepVO.cutout, _view);
_overlay.graphics.drawRect(_view.x + (_fteStepVO.cutout.x * _view.scaleX), _view.y +
(_fteStepVO.cutout.y * _view.scaleY), _fteStepVO.cutout.width * _view.scaleX,
_fteStepVO.cutout.
height *
_view.scaleY);
if (_trigger)
{
if (_rolloverListener)
addListener(_trigger, MouseEvent.ROLL_OVER, onTriggerClicked);
else
addListener(_trigger, MouseEvent.CLICK, onTriggerClicked);
} else
_triggerTimer.start();
} else
{
_overlay.graphics.drawRect(_fteStepVO.cutout.x, _fteStepVO.cutout.y,
_fteStepVO.cutout.width, _fteStepVO.cutout.height);
Application.STAGE.removeEventListener(Event.RESIZE, onResize);
Application.STAGE.addEventListener(Event.RESIZE, onResize, false, 0, true);
}
}

_overlay.graphics.endFill();
}

public
```

```
function showArrow():void
{
if (_arrow && _fteStepVO.arrowPosition != null)
{
_arrow.x = _fteStepVO.arrowPosition.x * ((_view) ? _view.scaleX : 1);
_arrow.y = _fteStepVO.arrowPosition.y * ((_view) ? _view.scaleY : 1);
if (_view)
{
_arrow.x += _view.x;
_arrow.y += _view.y;
}
_arrow.rotation = _fteStepVO.arrowRotation;
_arrow.visible = true;

if (!_view)
{
Application.STAGE.removeEventListener(Event.RESIZE, onResize);
Application.STAGE.addEventListener(Event.RESIZE, onResize, false, 0, true);
}
}
}

public function hideArrow():void
{
_arrow.visible = false;
}

public function rolloverTrigger():void
{
if (_trigger)
{
removeListener(_trigger, MouseEvent.CLICK, onTriggerClicked);
addListener(_trigger, MouseEvent.ROLL_OVER, onTriggerClicked);
} else
_rolloverListener = true;
}

override public function get typeUnique():Boolean { return false; }
override public function get type():String { return ViewEnum.ALERT; }

public function set clickToContinue( v:Boolean ):void { _clickToContinue = v; }

public function set dialogueView( v:FTEDialogueView ):void { _dialogueView = v; }
public function set fteStepVO( v:FTEStepVO ):void { _fteStepVO = v; }
public function set ignoreStoreOffset( v:Boolean ):void { _ignoreStoreOffset = v; }
public function set trigger( v:DisplayObject ):void
{
if (_trigger && _trigger.filters.length > 0)
_trigger.filters = [];

_trigger
```

```
= v;
}

public function set view( v:* ):void { _view = v; }
public function set viewAlreadyExisted( v:Boolean ):void { _viewAlreadyExisted = v; }

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
super.destroy();

_arrow = null;
_fteStepVO = null;
_ignoreStoreOffset = false;
_overlay.graphics.clear();
_overlay = null;
_rolloverListener = false;
_trigger = null;
_triggerTimer.stop();
_triggerTimer.removeEventListener(TimerEvent.TIMER_COMPLETE, showCutout);
_triggerTimer = null;
if (_view && _view.effects != null)
{
ViewEffects(_view.effects).removeInListener(onEffectDoneIn);
}
_view = null;
_dialogueView = null;
Application.STAGE.removeEventListener(Event.RESIZE, onResize);
}
}
}
```

File 664: igw\com\ui\alert\InputAlertView.as

```
package com.ui.alert
{
import com.Application;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import
```

```
org.adobe.utils.StringUtil;

public class InputAlertView extends AlertView
{
private var _inputBG:Sprite;
private var _inputText:Label;

private var _onCloseUseBtnTwo:Boolean;

override public function setUp( args:Array ):void
{

var restrict:String = args[12];
var frameBGClass:Class = Class(getDefinitionByName(('TextInputFieldMC')));
_inputBG = Sprite(new frameBGClass());

_inputText = new Label(20, 0xffffff, 200, 25);
_inputText.constrictTextToSize = false;
_inputText.letterSpacing = .8;
_inputText.align = TextFormatAlign.LEFT;
_inputText.addLabelColor(0xbdfefd, 0x000000);
_inputText.maxChars = args[9];
_inputText.text = args[10];
_inputText.allowInput = true;
_inputText.clearOnFocusIn = args[11];
if (restrict != '')
_inputText.restrict = restrict;
_inputText.addEventListener(Event.CHANGE, onTextChanged, false, 0, true);
_onCloseUseBtnTwo = args[8];

super.setUp(args);
addChild(_inputBG);
addChild(_inputText);
}

override protected function layout():void
{
super.layout();

var height:Number;
if (_btnOne != null)
{
if (_btnTwo == null)
_btnOne.x = 150;
else
_btnOne.x = 30;

_btnOne.y = _bg.height - 60;
height = _btnOne.y + _btnOne.height + 10
} else
height
```

```
= _inputBG.y + _inputBG.height + 10

if (_btnTwo != null)
{
_btnTwo.x = _bg.width - (_btnTwo.width + 20);
_btnTwo.y = _bg.height - 60;
}

_inputBG.x = 20;
_inputBG.y = _btnOne.y - _inputBG.height - 10;

_inputText.width = _inputBG.width - 40;
_inputText.x = _inputBG.x + 15;
_inputText.y = _inputBG.y + 2;

_bodyText.setSize(_bg.width - 40, _inputBG.y - (_viewName.y + _viewName.height));

Application.STAGE.focus = _inputText;
_inputText.setSelection(0, _inputText.length);
}

private function onTextChanged( e:Event ):void
{
var currentTextLen:uint = e.currentTarget.length;
var btnToUse:BitmapButton;
var btnFunctionToUse:Function;
if (!_onCloseUseBtnTwo)
{
btnToUse = _btnOne;
btnFunctionToUse = onBtnOneClick;
} else
{
btnToUse = _btnTwo;
btnFunctionToUse = onBtnTwoClick;
}

if (currentTextLen != 0 && !btnToUse.enabled)
{
btnToUse.enabled = true;
btnToUse.addEventListener(MouseEvent.CLICK, btnFunctionToUse, false, 0, true);
} else if (currentTextLen == 0 && btnToUse.enabled == true)
{
btnToUse.enabled = false;
btnToUse.removeEventListener(MouseEvent.CLICK, btnFunctionToUse);
}
}

override protected function onEnterPress( keyCode:uint ):void
{
onClose(null);
}
```

```
override protected function btnOnePress():void
{
var text:String = StringUtil.escapeHTML(_inputText.text);
_btnOneArgs = new Array(text);
super.btnOnePress();
}

override protected function btnTwoPress():void
{
var text:String = StringUtil.escapeHTML(_inputText.text);
_btnTwoArgs = new Array(text);
super.btnTwoPress();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function destroy():void
{
_inputBG = null;

_inputText.removeEventListener(Event.CHANGE, onTextChanged);
_inputText.destroy();
_inputText = null;

super.destroy()
}
}
}

-------------------------------------
File 665: igw\com\ui\core\ButtonPrototype.as
package com.ui.core
{
import com.enum.ui.ButtonEnum;

public class ButtonPrototype
{
public var type:String;
public var text:String;
public var callback:Function;
public var args:Array = [];
public var doClose:Boolean;

public function ButtonPrototype( text:String, callback:Function = null, args:Array = null,
doClose:Boolean = true, type:String = 'BtnBlueA' )
{
this.text = text;
this.callback
```

= callback;
if (args)
this.args = args;
this.doClose = doClose;
this.type = type;
}
}
}

File 666: igw\com\ui\core\DefaultWindowBG.as
ï»¿package com.ui.core
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.events.MouseEvent;
import flash.events.Event;

public class DefaultWindowBG extends Sprite
{
private var _bg:ScaleBitmap;
private var _closeButton:BitmapButton;
private var _headerClose:ScaleBitmap;
private var _headerSide:ScaleBitmap;
private var _headerTop:ScaleBitmap;
private var _label:Label;

public function DefaultWindowBG()
{
_bg = UIFactory.getScaleBitmap(PanelEnum.WINDOW);
_closeButton = UIFactory.getButton(ButtonEnum.CLOSE);
_headerClose = UIFactory.getScaleBitmap(PanelEnum.WINDOW_X_HEADER);
_headerSide = UIFactory.getScaleBitmap(PanelEnum.WINDOW_SIDE_HEADER);
_headerTop = UIFactory.getScaleBitmap(PanelEnum.WINDOW_HEADER);

addChild(_bg);
addChild(_headerSide);
addChild(_headerTop);
addChild(_headerClose);
addChild(_closeButton);

layout();
}

```
public function getBkgdScale9Grid():Rectangle
{
return _bg ? _bg.scale9Grid : null;
}

public function addTitle( text:String, width:Number ):void
{
_headerTop.setSize(0, 0);
_headerTop.setSize(width + _headerTop.src.width, 0);
if (_label)
{
removeChild(_label);
_label = UIFactory.destroyLabel(_label);
}
_label = UIFactory.getLabel(LabelEnum.H1, _headerTop.width - 196, 40, 25, 2);
_label.align = TextFormatAlign.LEFT;
_label.constrictTextToSize = true;
_label.text = text;
addChild(_label);
layout();
}

public function setBGSize( width:Number, height:Number ):void
{
_bg.setSize(0, 0);
_bg.setSize(width, height);
layout();
}

public function setSideSize( width:Number, height:Number ):void
{
_headerSide.setSize(0, 0);
_headerSide.setSize(width, height);
layout();
}

private function layout():void
{
_bg.x = 11;
_bg.y = _headerTop.height;

_headerSide.x = 0;
_headerSide.y = _headerTop.height;

_headerTop.x = 0;
_headerTop.y = 0;

_headerClose.x = _bg.width - 113 + _bg.x;
_headerClose.y
```

```
= 20;

_closeButton.x = _headerClose.x + 60;
_closeButton.y = _headerClose.y + 1;

if(CONFIG::IS_MOBILE){
_closeButton.scaleX = _closeButton.scaleY = 4;
_closeButton.y -= 64;
}
}

public function get bg():ScaleBitmap { return _bg; }
public function get closeButton():BitmapButton { return _closeButton; }
public function set titleFontSize( v:int ):void { _label.fontSize = v; }

public function destroy():void
{
x = y = 0;
if (_label)
{
removeChild(_label);
_label = UIFactory.destroyLabel(_label);
}
}
}
}

-------------------------------------
File 667: igw\com\ui\core\ScaleBitmap.as
/**
*
* ScaleBitmap
*
* @version 1.1
* @author Didier BRUN - http://www.bytearray.org
*
* @version 1.2.1
* @author Alexandre LEGOUT - http://blog.lalex.com
*
* @version 1.2.2
* @author Pleh
*
* Project page : http://www.bytearray.org/?p=118
*
*/

package com.ui.core
{
import flash.display.Bitmap;
import flash.display.BitmapData;
import
```

```
flash.geom.Matrix;
import flash.geom.Rectangle;

public class ScaleBitmap extends Bitmap
{
protected var _height:Number = 0;
protected var _matrix:Matrix;
protected var _originalBitmap:BitmapData;
protected var _scaled:Boolean;
protected var _scale9Grid:Rectangle = null;
protected var _width:Number = 0;

public function ScaleBitmap( bmpData:BitmapData = null, pixelSnapping:String = "auto",
smoothing:Boolean = false )
{
_matrix = new Matrix();
_scaled = false;

// original bitmap
if (bmpData)
_originalBitmap = bmpData;

// super constructor
super(_originalBitmap, pixelSnapping, smoothing);
}

// ----------------------------------------------
//
// ---o public methods
//
// ----------------------------------------------

/**
* setter bitmapData
*/
override public function set bitmapData( bmpData:BitmapData ):void
{
if (_scaled && super.bitmapData)
super.bitmapData.dispose();
_originalBitmap = bmpData;
_scaled = false;
if (bmpData)
{
if (_scale9Grid != null)
{
if (!validGrid(_scale9Grid))
{
_scale9Grid = null;
}
setSize(_width, _height);
}
```

```
else
{
super.bitmapData = bmpData;
}
}
}

public function get src():BitmapData
{
return _originalBitmap;
}

/**
* setter width
*/
override public function set width( w:Number ):void
{
_width = w;
setSize(w, _height);
}

/**
* setter height
*/
override public function set height( h:Number ):void
{
_height = h;
setSize(_width, h);
}

/**
* set scale9Grid
*/
override public function set scale9Grid( r:Rectangle ):void
{
// Check if the given grid is different from the current one
if ((_scale9Grid == null && r != null) || (_scale9Grid != null && !_scale9Grid.equals(r)))
{
if (r == null)
{
// If deleting scalee9Grid, restore the original bitmap
// then resize it (streched) to the previously set dimensions
_scale9Grid = null;
bitmapData = _originalBitmap;
setSize(_width, _height);
} else
{
if (!validGrid(r))
{
throw(new Error("#001 - The _scale9Grid does not match the original BitmapData"));
return;
```

```
}

_scale9Grid = r.clone();
resizeBitmap(width, height);
scaleX = 1;
scaleY = 1;
}
}
}

private function validGrid( r:Rectangle ):Boolean
{
return r.right <= _originalBitmap.width && r.bottom <= _originalBitmap.height;
}

/**
* get scale9Grid
*/
override public function get scale9Grid():Rectangle
{
return _scale9Grid;
}

/**
* setSize
*/
public function setSize( w:Number, h:Number ):void
{
if (_scale9Grid == null)
{
if (w > 0)
super.width = w;
if (h > 0)
super.height = h;
} else
{
w = Math.max(w, _originalBitmap.width);
h = Math.max(h, _originalBitmap.height);
if (w > 0 && h > 0)
{
_width = w;
_height = h;
resizeBitmap(w, h);
}
}
}

/**
* get original bitmap
*/
```

```
public function getOriginalBitmapData():BitmapData
{
return _originalBitmap;
}

// ----------------------------------------------
//
// ---o protected methods
//
// ----------------------------------------------

/**
* resize bitmap
*/
protected function resizeBitmap( w:Number, h:Number ):void
{
_scaled = true;
var bmpData:BitmapData = new BitmapData(w, h, true, 0x00000000);

var rows:Array = [0, _scale9Grid.top, _scale9Grid.bottom, _originalBitmap.height];
var cols:Array = [0, _scale9Grid.left, _scale9Grid.right, _originalBitmap.width];

var dRows:Array = [0, _scale9Grid.top, h - (_originalBitmap.height - _scale9Grid.bottom), h];
var dCols:Array = [0, _scale9Grid.left, w - (_originalBitmap.width - _scale9Grid.right), w];

var origin:Rectangle;
var draw:Rectangle;

for (var cx:int = 0; cx < 3; cx++)
{
for (var cy:int = 0; cy < 3; cy++)
{
origin = new Rectangle(cols[cx], rows[cy], cols[cx + 1] - cols[cx], rows[cy + 1] - rows[cy]);
draw = new Rectangle(dCols[cx], dRows[cy], dCols[cx + 1] - dCols[cx], dRows[cy + 1] -
dRows[cy]);
_matrix.identity();
_matrix.a = draw.width / origin.width;
_matrix.d = draw.height / origin.height;
_matrix.tx = draw.x - origin.x * _matrix.a;
_matrix.ty = draw.y - origin.y * _matrix.d;
bmpData.draw(_originalBitmap, _matrix, null, null, draw, smoothing);
}
}
super.bitmapData = bmpData;
}

public function destroy():void
{
if (_scaled && super.bitmapData)
super.bitmapData.dispose();
```

```
filters = [];
_matrix.identity();
_width = _height = 0;
_originalBitmap = null;
_scaled = false;
_scale9Grid = null;
}
}
}

-----------------------------------
File 668: igw\com\ui\core\View.as
package com.ui.core
{
import com.Application;
import com.controller.keyboard.KeyboardController;
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.event.ToastEvent;
import com.model.prototype.IPrototype;
import com.presenter.IImperiumPresenter;
import com.ui.alert.ConfirmationView;
import com.ui.alert.InputAlertView;
import com.ui.core.effects.EffectFactory;
import com.ui.core.effects.GenericMoveEffect;
import com.ui.core.effects.ViewEffects;

import flash.display.Sprite;
import flash.events.IEventDispatcher;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.parade.core.IView;
import org.parade.core.IViewFactory;
import org.parade.enum.ViewEnum;
import org.robotlegs.extensions.localEventMap.api.IEventMap;
import org.robotlegs.extensions.localEventMap.impl.EventMap;
import org.shared.ObjectPool;

public class View extends Sprite implements IView
{
protected var _effects:ViewEffects;
protected var _eventMap:IEventMap = new EventMap(null);
protected var _keyboard:KeyboardController;
protected var _mute:Boolean = false;
protected var _presenter:IImperiumPresenter;
protected var _viewFactory:IViewFactory;

[PostConstruct]
public
```

```
function init():void
{
// PR: Removing the creation of the event map that was here. No longer nulling eventMap out on
destroy
// all events will be removed from the map so when object pooling we can reuse the existing one
without having to create a new one
_presenter && _presenter.addStateListener(onStateChange) && _presenter.highfive();
addListener(this, MouseEvent.RIGHT_MOUSE_DOWN, onRightMouse);
addListener(this, MouseEvent.MOUSE_DOWN, onMouseDown);
addListener(this, MouseEvent.MOUSE_WHEEL, onMouseDown);
_effects = ObjectPool.get(ViewEffects);
_effects.addInListener(effectsDoneIn);
_effects.addOutListener(effectsDoneOut);
}

protected function onMouseDown( e:MouseEvent ):void
{
e.stopPropagation();
}

protected function onRightMouse( e:MouseEvent ):void
{
e.stopPropagation();
}

public function onEscapePressed():void
{
if ((type == ViewEnum.MODAL || type == ViewEnum.HOVER || type == ViewEnum.ALERT) &&
_presenter && !_presenter.inFTE)
destroy();
}

protected function addHitArea( hitArea:Sprite = null ):void
{
if (hitArea == null)
{
hitArea = new Sprite();
hitArea.mouseEnabled = false;
}
this.hitArea = hitArea;
}

protected function onStateChange( state:String ):void { destroy(); }

protected function onClose( e:MouseEvent = null ):void { destroy(); }

//- EFFECT CONTROLS ------------------------------------------------------------------------

protected function addEffects():void
{
_effects.addEffect(EffectFactory.resizeEffect());
```

```
_effects.addEffect(EffectFactory.simpleBackingEffect(.75, 0, 0));
_effects.addEffect(EffectFactory.genericMoveEffect(GenericMoveEffect.UP,
GenericMoveEffect.UP, .5, .3));
}

protected function effectsIN():void { _effects && _effects.effectsIn(this); }
protected function effectsDoneIn():void {}
protected function effectsOUT():void { _effects && _effects.effectsOut(this); }
protected function effectsDoneOut():void
{
if (_viewFactory)
{
ObjectPool.give(_effects);
_effects = null;

while (numChildren > 0)
removeChildAt(0);

if (this.parent != null)
this.parent.removeChild(this);

_viewFactory.destroyView(this);
_viewFactory = null;
}
}

//- ----------------------------------------------------------------------------------------

protected function showView( view:Class, notify:Boolean = true ):IView
{
var nview:IView = _viewFactory.createView(view);
if (notify)
_viewFactory.notify(nview);

if (!_mute)
SoundController.instance.playSound(AudioEnum.AFX_UI_WINDOW_OPEN, 0.5);

return nview;
}

protected function showInputAlert( alertTitle:String, alertBody:String, btnOneText:String,
btnOneCallback:Function, btnOneArgs:Array, btnTwoText:String, btnTwoCallback:Function,
btnTwoArgs:Array,
onCloseUseBtnTwo:Boolean = false, maxCharacters:int = 12, defaultInputText:String = '',
clearInputOnFocus:Boolean = false, restrict:String = '', notify:Boolean =
true ):IView
{
return _viewFactory.createAlert(alertTitle, alertBody, btnOneText, btnOneCallback, btnOneArgs,
btnTwoText, btnTwoCallback, btnTwoArgs, onCloseUseBtnTwo, maxCharacters,
defaultInputText,
```

```
clearInputOnFocus,
restrict,
notify, InputAlertView);
}

protected function showConfirmation( title:String, body:String, buttons:Vector.<ButtonPrototype>
):IView
{
var view:ConfirmationView = ConfirmationView(_viewFactory.createView(ConfirmationView));
view.setup(title, body, buttons);
_viewFactory.notify(view);
return view;
}

protected function showToast( type:Object, prototype:IPrototype = null, ... strings ):void
{
if (!_presenter)
return;

var toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = type;
toastEvent.prototype = prototype;
if (strings)
toastEvent.addStringsFromArray(strings);
_presenter.dispatch(toastEvent);
}

public function addListener( dispatcher:IEventDispatcher, type:String, listener:Function,
eventClass:Class = null, useCapture:Boolean = false, priority:int = 0,
useWeakReference:Boolean = true ):void
{
_eventMap.mapListener(dispatcher, type, listener, eventClass, useCapture, priority,
useWeakReference);
}

public function removeListener( dispatcher:IEventDispatcher, type:String, listener:Function ):void
{
_eventMap.unmapListener(dispatcher, type, listener);
}

public function get bounds():Rectangle { if (!parent) return getBounds(Application.STAGE);
return getBounds(parent); }
public function get effects():ViewEffects { return _effects; }
public function get type():String { return ViewEnum.MODAL; }
public function get typeUnique():Boolean { return false; }
public function get screenshotBlocker():Boolean {return false;}

[Inject]
public function set keyboard( value:KeyboardController ):void { _keyboard = value; }
[Inject]
public
```

```
function set viewFactory( value:IViewFactory ):void { _viewFactory = value; }

public function destroy():void
{
_eventMap && _eventMap.unmapListeners();

_presenter && _presenter.removeStateListener(onStateChange) && _presenter.shun();
_presenter = null;

if (!_mute)
SoundController.instance.playSound(AudioEnum.AFX_UI_WINDOW_CLOSE, 0.5);

_keyboard = null;
effectsOUT();

_mute = false;
}
}
}

-----------------------------------
File 669: igw\com\ui\core\ViewFactory.as
package com.ui.core
{
import com.Application;
import com.ui.alert.AlertView;
import com.event.PaywallEvent;
import com.service.ExternalInterfaceAPI;

import flash.events.IEventDispatcher;

import org.parade.core.IView;
import org.parade.core.IViewFactory;
import org.parade.core.ViewEvent;
import com.event.ServerEvent;

public class ViewFactory implements IViewFactory
{
private var _eventDispatcher:IEventDispatcher;

public function createView( targetClass:Class ):IView
{
var view:IView;
switch (targetClass)
{
default:
view = new targetClass();
break;
}
return view;
}
```

```
public function createAlert( alertTitle:String, alertBody:String, btnOneText:String,
btnOneCallback:Function, btnOneArgs:Array, btnTwoText:String, btnTwoCallback:Function,
btnTwoArgs:Array,
onCloseUseBtnTwo:Boolean = false, maxCharacters:int = 12, defaultInputText:String = '',
clearInputOnFocus:Boolean = false, restrict:String = '', shouldNotify:Boolean =
true, view:Class =
null ):IView
{
if (view == null)
view = AlertView;

var nAlertview:AlertView = view(createView(view));
var alertArgs:Array = new Array(alertTitle, alertBody, btnOneText, btnOneCallback, btnOneArgs,
btnTwoText, btnTwoCallback, btnTwoArgs, onCloseUseBtnTwo, maxCharacters,
defaultInputText,
clearInputOnFocus);
nAlertview.setUp(alertArgs);
if (shouldNotify)
notify(nAlertview);

return nAlertview;
}

public function notify( view:IView ):void
{
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.SHOW_VIEW);
viewEvent.targetView = view;
_eventDispatcher.dispatchEvent(viewEvent);
}

public function destroyView( targetView:IView ):void
{
var viewEvent:ViewEvent = new ViewEvent(ViewEvent.DESTROY_VIEW);
viewEvent.targetView = targetView;
_eventDispatcher.dispatchEvent(viewEvent);

Application.STAGE.focus = Application.STAGE;
}

public function openPayment():void
{
ExternalInterfaceAPI.logConsole("Open Payment");
if (Application.NETWORK == Application.NETWORK_KONGREGATE)
{
ExternalInterfaceAPI.logConsole("Kongregate payments");
var paywall:PaywallEvent = new PaywallEvent(PaywallEvent.GET_PAYWALL);
_eventDispatcher.dispatchEvent(paywall);
}
else
```

```
if (Application.NETWORK == Application.NETWORK_FACEBOOK)
{
ExternalInterfaceAPI.logConsole("Facebook payments");
var paywall:PaywallEvent = new PaywallEvent(PaywallEvent.GET_PAYWALL);
_eventDispatcher.dispatchEvent(paywall);
}
else if (Application.NETWORK == Application.NETWORK_XSOLLA)
{
ExternalInterfaceAPI.logConsole("Xsolla payments");
var serverEvent:ServerEvent
serverEvent = new ServerEvent(ServerEvent.OPEN_PAYMENT);
_eventDispatcher.dispatchEvent(serverEvent);
//_viewFactory.openPayment();
//ExternalInterfaceAPI.popPayWall();
}
else if (Application.NETWORK == Application.NETWORK_GUEST)
{
ExternalInterfaceAPI.logConsole("Guest Payment Restriction");
var serverEvent:ServerEvent
serverEvent = new ServerEvent(ServerEvent.GUEST_RESTRICTION);
_eventDispatcher.dispatchEvent(serverEvent);
}
else if (Application.NETWORK == Application.NETWORK_STEAM)
{
//todo steam payments
}
//var serverEvent:ServerEvent
//serverEvent = new ServerEvent(ServerEvent.OPEN_PAYMENT);
//_eventDispatcher.dispatchEvent(serverEvent);
}

[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
}
}

-------------------------------------
File 670: igw\com\ui\core\ViewStack.as
package com.ui.core
{
import com.Application;
import com.event.StateEvent;
import com.presenter.shared.IUIPresenter;
import com.util.TimeLog;

import flash.display.DisplayObject;
import flash.display.DisplayObjectContainer;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.IEventDispatcher;
import
```

```
flash.geom.Rectangle;
import flash.system.Capabilities;

import org.ash.tick.ITickProvider;
import org.console.Cc;
import org.parade.core.IView;
import org.parade.core.IViewStack;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.starling.core.Starling;
import org.starling.display.DisplayObject;
import org.starling.display.Sprite;
import org.starling.events.Event;

public class ViewStack implements IViewStack
{
protected const _alert:flash.display.Sprite = new flash.display.Sprite();
protected const _console:flash.display.Sprite = new flash.display.Sprite();
protected const _error:flash.display.Sprite = new flash.display.Sprite();
protected const _hover:flash.display.Sprite = new flash.display.Sprite();
protected const _modal:flash.display.Sprite = new flash.display.Sprite();
protected const _ui:flash.display.Sprite = new flash.display.Sprite();
protected const _game:flash.display.Sprite = new flash.display.Sprite();
protected const _game3D:org.starling.display.Sprite = new org.starling.display.Sprite();
protected const _maxAmplitude:int = 10;

protected var _contextView:flash.display.DisplayObjectContainer;
protected var _backgroundLayer:flash.display.Sprite;
protected var _background3DLayer:org.starling.display.Sprite;
protected var _gameLayer:flash.display.Sprite;
protected var _game3DLayer:org.starling.display.Sprite;
protected var _presenter:IUIPresenter;

protected var _amplitude:Number;
protected var _direction:int;
protected var _eventDispatcher:IEventDispatcher;
protected var _frameTickProvider:ITickProvider;
protected var _initialAmplitude:Number;
protected var _speed:Number;
protected var _starling:Starling;
protected var _time:Number;
protected var _totalTime:Number;
protected var _wave:int;

[PostConstruct]
public function init():void
{
_contextView.stage.addEventListener(flash.events.Event.RESIZE, onResize);
_contextView.addChild(_ui);
_contextView.addChild(_modal);
_contextView.addChild(_alert);
```

```
_contextView.addChild(_hover);
_contextView.addChild(_error);
_contextView.addChild(_console);

Cc.config.commandLineAutoCompleteEnabled = true;
Cc.config.commandLineAllowed = true;
Cc.commandLine = true;
Cc.startOnStage(_console, "~");

onResize(null);
}

private function onResize( e:flash.events.Event ):void
{
//need a check here because in IE stagewidth / stageheight may be 0. wait for a resize event to
tell us width / height is available
if (DeviceMetrics.WIDTH_PIXELS > 0)
{
_contextView.stage.removeEventListener(flash.events.Event.RESIZE, onResize);
_starling = new Starling(org.starling.display.Sprite, _contextView.stage, new Rectangle(0, 0,
DeviceMetrics.WIDTH_PIXELS, DeviceMetrics.HEIGHT_PIXELS)); //, null, "auto",
Context3DProfile.BASELINE); //, null, "software");
//_starling = new Starling(org.starling.display.Sprite, _contextView.stage, new Rectangle(0, 0,
DeviceMetrics.WIDTH_PIXELS, DeviceMetrics.HEIGHT_PIXELS, null, "auto",
"baselineExtended"); //, null, "software");
_starling.addEventListener(org.starling.events.Event.ROOT_CREATED, addStage3DLayers);
_starling.start();

clearValues();
}
}

public function addView( view:IView ):void
{
addToLayer(view, view.type);
}

public function addToLayer( object:Object, layer:String ):void
{
switch (layer)
{
case ViewEnum.ALERT:
_alert.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.ERROR:
_error.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.HOVER:
_hover.addChild(flash.display.DisplayObject(object));
break;
```

```
case ViewEnum.MODAL:
_modal.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.UI:
_ui.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.GAME:
if (Application.STARLING_ENABLED)
_game3D.addChildAt(org.starling.display.DisplayObject(object), 0);
else
_game.addChildAt(flash.display.DisplayObject(object), 0);
break;
case ViewEnum.BACKGROUND_LAYER:
if (Application.STARLING_ENABLED)
_background3DLayer.addChild(org.starling.display.DisplayObject(object));
else
_backgroundLayer.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.GAME_LAYER:
if (Application.STARLING_ENABLED)
_game3DLayer.addChild(org.starling.display.DisplayObject(object));
else
_gameLayer.addChild(flash.display.DisplayObject(object));
break;
}
}

public function getLayer( layer:String ):*
{
switch (layer)
{
case ViewEnum.ALERT:
return _alert;
break;
case ViewEnum.ERROR:
return _error;
break;
case ViewEnum.HOVER:
return _hover;
break;
case ViewEnum.MODAL:
return _modal;
break;
case ViewEnum.UI:
return _ui;
break;
case ViewEnum.GAME:
if (Application.STARLING_ENABLED)
return _game3D;
else
```

```
return _game;
break;
case ViewEnum.BACKGROUND_LAYER:
if (Application.STARLING_ENABLED)
return _background3DLayer;
else
return _backgroundLayer;
break;
case ViewEnum.GAME_LAYER:
if (Application.STARLING_ENABLED)
return _game3DLayer;
else
return _gameLayer;
break;
}
}

public function shake( amplitude:Number = 1, time:int = 1, speed:int = 10, direction:int = 3 ):void
{
_wave = 0;

if (amplitude < 0)
amplitude = 0;
_amplitude += amplitude;
if (_amplitude > _maxAmplitude)
_amplitude = _maxAmplitude;

_initialAmplitude = _amplitude;
_time = 0;
_totalTime = time;
_speed = speed;
_direction = direction;
}

public function update( time:Number ):void
{
if (_totalTime == 0)
return;

_wave += _speed / _maxAmplitude;
_amplitude = _initialAmplitude - _initialAmplitude / _totalTime * _time;

if (_time < _totalTime)
{

switch (_direction)
{
case 1:
_game3D.x = Math.cos(_wave) * _amplitude;
break;
```

```
case 2:
_game3D.y = Math.sin(_wave / 1.5) * _amplitude;
break;

case 3:
_game3D.x = Math.cos(_wave) * _amplitude;
_game3D.y = Math.sin(_wave / 1.5) * _amplitude;
break;
}
} else
{
_game3D.x = _game3D.y = 0;
clearValues();
}

_time += time;
}

public function clearLayer( layer:String ):void
{
var viewLayer:*;
switch (layer)
{
case ViewEnum.ALERT:
case ViewEnum.ERROR:
case ViewEnum.HOVER:
case ViewEnum.MODAL:
case ViewEnum.UI:
break;
case ViewEnum.GAME:
if (Application.STARLING_ENABLED)
viewLayer = _game3D;
else
viewLayer = _game;
break;
case ViewEnum.BACKGROUND_LAYER:
if (Application.STARLING_ENABLED)
viewLayer = _background3DLayer;
else
viewLayer = _backgroundLayer;
break;
case ViewEnum.GAME_LAYER:
if (Application.STARLING_ENABLED)
viewLayer = _game3DLayer;
else
viewLayer = _gameLayer;
break;
}
while
```

```
(viewLayer.numChildren > 0)
viewLayer.removeChildAt(0);
}

public function clearValues():void
{
_wave = _amplitude = _time = _speed = _direction = _totalTime = 0;
}

private function addStage3DLayers( e:org.starling.events.Event ):void
{
Application.STARLING_ENABLED =
Starling.context.driverInfo.toLowerCase().indexOf("software") == -1;
if (Application.STARLING_ENABLED)
{
_starling.stage.addChild(_game3D);
_background3DLayer = new org.starling.display.Sprite();
_background3DLayer.touchable = false;
_game3DLayer = new org.starling.display.Sprite();
_game3DLayer.touchable = false;
_game3D.addChild(_background3DLayer);
_game3D.addChild(_game3DLayer);
_game3D.alpha = 0.999;
_starling.addEventListener(org.starling.events.Event.CONTEXT3D_CREATE,
onContextCreated);
//if (CONFIG::DEBUG == true)
//_starling.showStatsAt("center", "bottom");
} else
{
_starling.stop();
_starling.dispose();
_starling = null;
_contextView.addChildAt(_game, 0);
_backgroundLayer = new flash.display.Sprite();
_backgroundLayer.mouseChildren = _backgroundLayer.mouseEnabled = false;
_gameLayer = new flash.display.Sprite();
_gameLayer.mouseChildren = _gameLayer.mouseEnabled = false;
_game.addChild(_backgroundLayer);
_game.addChild(_gameLayer);
}
logSpecs();
}

private function onContextCreated( e:org.starling.events.Event ):void
{
_eventDispatcher.dispatchEvent(new StateEvent(StateEvent.LOST_CONTEXT));
}

private static function logSpecs():void
{
//
```

```
Get the playerâ€™s version by using the flash.system.Capabilities class.
var versionNumber:String = Capabilities.version;
var log:String = "[SPECS] CPU:" + Capabilities.cpuArchitecture + " -- OS:" + Capabilities.os;
var versionArray:Array = versionNumber.split(",");
var platformAndVersion:Array = versionArray[0].split(" ");

log += " -- FLASH_PLAYER:" + (int(platformAndVersion[1]) + "." + int(versionArray[1]));
log += " -- RESOLUTION:" + Capabilities.screenResolutionX + "x" +
Capabilities.screenResolutionY;
log += " -- GPU:" + (Application.STARLING_ENABLED ? "true" : "false");
TimeLog.addLog(log);
}

[Inject]
public function set contextView( value:flash.display.DisplayObjectContainer ):void {
_contextView = value; }
[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
[Inject]
public function set frameTickProvider( value:ITickProvider ):void { _frameTickProvider = value; }
}
}



-------------------------------------
File 671: igw\com\ui\core\ViewStack3D.as
package com.ui.core
{
import com.Application;
import com.event.StateEvent;
import com.presenter.shared.IUIPresenter;
import com.util.TimeLog;

import flash.display.DisplayObject;
import flash.display.DisplayObjectContainer;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.IEventDispatcher;
import flash.geom.Vector3D;
import flash.system.Capabilities;

import org.ash.tick.ITickProvider;
import org.away3d.cameras.lenses.OrthographicLens;
import org.away3d.containers.ObjectContainer3D;
import org.away3d.containers.View3D;
import org.away3d.core.managers.Stage3DManager;
import org.away3d.core.managers.Stage3DProxy;
import org.away3d.events.Stage3DEvent;
import org.away3d.lights.DirectionalLight;
import
```

```
org.away3d.loaders.parsers.Parsers;
import org.away3d.materials.lightpickers.StaticLightPicker;
import org.console.Cc;
import org.parade.core.IView;
import org.parade.core.IViewStack;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.starling.core.Starling;
import org.starling.display.DisplayObject;
import org.starling.display.Sprite;
import org.starling.events.Event;

public class ViewStack3D implements IViewStack
{
public static var lightPicker:StaticLightPicker;

protected const _alert:flash.display.Sprite = new flash.display.Sprite();
protected const _console:flash.display.Sprite = new flash.display.Sprite();
protected const _error:flash.display.Sprite = new flash.display.Sprite();
protected const _hover:flash.display.Sprite = new flash.display.Sprite();
protected const _modal:flash.display.Sprite = new flash.display.Sprite();
protected const _ui:flash.display.Sprite = new flash.display.Sprite();
protected const _game:flash.display.Sprite = new flash.display.Sprite();
protected const _game3D:org.starling.display.Sprite = new org.starling.display.Sprite();
protected const _maxAmplitude:int = 10;

protected var _contextView:flash.display.DisplayObjectContainer;
protected var _backgroundLayer:flash.display.Sprite;
protected var _background3DLayer:org.starling.display.Sprite;
protected var _gameLayer:flash.display.Sprite;
protected var _game3DLayer:org.starling.display.Sprite;
protected var _presenter:IUIPresenter;

protected var _amplitude:Number;
protected var _direction:int;
protected var _eventDispatcher:IEventDispatcher;
protected var _frameTickProvider:ITickProvider;
protected var _initialAmplitude:Number;
protected var _speed:Number;
protected var _starling:Starling;
protected var _time:Number;
protected var _totalTime:Number;
protected var _wave:int;

protected var _away3D:View3D;
protected var _stage3DManager:Stage3DManager;
protected var _stage3DProxy:Stage3DProxy;

[PostConstruct]
public function init():void
{
```

```
_contextView.stage.addEventListener(flash.events.Event.RESIZE, onResize);
_contextView.addChild(_ui);
_contextView.addChild(_modal);
_contextView.addChild(_alert);
_contextView.addChild(_hover);
_contextView.addChild(_error);
_contextView.addChild(_console);

Cc.config.commandLineAutoCompleteEnabled = true;
Cc.config.commandLineAllowed = true;
Cc.commandLine = true;
Cc.startOnStage(_console, "~");

onResize(null);
}

private function onResize( e:flash.events.Event ):void
{
//need a check here because in IE stagewidth / stageheight may be 0. wait for a resize event to
tell us width / height is available
if (DeviceMetrics.WIDTH_PIXELS > 0)
{
_contextView.stage.removeEventListener(flash.events.Event.RESIZE, onResize);
_stage3DManager = Stage3DManager.getInstance(_contextView.stage);

_stage3DProxy = _stage3DManager.getFreeStage3DProxy();
_stage3DProxy.addEventListener(Stage3DEvent.CONTEXT3D_CREATED,
addStage3DLayers);
_stage3DProxy.antiAlias = 8;
_stage3DProxy.color = 0x00ff00;

clearValues();
}
}

public function addView( view:IView ):void
{
addToLayer(view, view.type);
}

public function addToLayer( object:Object, layer:String ):void
{
switch (layer)
{
case ViewEnum.ALERT:
_alert.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.ERROR:
_error.addChild(flash.display.DisplayObject(object));
break;
```

```
case ViewEnum.HOVER:
_hover.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.MODAL:
_modal.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.UI:
_ui.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.GAME:
if (Application.STARLING_ENABLED)
_game3D.addChildAt(org.starling.display.DisplayObject(object), 0);
else
_game.addChildAt(flash.display.DisplayObject(object), 0);
break;
case ViewEnum.BACKGROUND_LAYER:
if (Application.STARLING_ENABLED)
_background3DLayer.addChild(org.starling.display.DisplayObject(object));
else
_backgroundLayer.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.GAME_LAYER:
if (Application.STARLING_ENABLED)
_game3DLayer.addChild(org.starling.display.DisplayObject(object));
else
_gameLayer.addChild(flash.display.DisplayObject(object));
break;
case ViewEnum.AWAY3D_LAYER:
_away3D.scene.addChild(ObjectContainer3D(object));
break;
}
}

public function getLayer( layer:String ):*
{
switch (layer)
{
case ViewEnum.ALERT:
return _alert;
break;
case ViewEnum.ERROR:
return _error;
break;
case ViewEnum.HOVER:
return _hover;
break;
case ViewEnum.MODAL:
return _modal;
break;
case
```

```
ViewEnum.UI:
return _ui;
break;
case ViewEnum.GAME:
if (Application.STARLING_ENABLED)
return _game3D;
else
return _game;
break;
case ViewEnum.BACKGROUND_LAYER:
if (Application.STARLING_ENABLED)
return _background3DLayer;
else
return _backgroundLayer;
break;
case ViewEnum.GAME_LAYER:
if (Application.STARLING_ENABLED)
return _game3DLayer;
else
return _gameLayer;
break;
case ViewEnum.AWAY3D_LAYER:
return _away3D.scene;
break;
}
}

public function shake( amplitude:Number = 1, time:int = 1, speed:int = 10, direction:int = 3 ):void
{
_wave = 0;

if (amplitude < 0)
amplitude = 0;
_amplitude += amplitude;
if (_amplitude > _maxAmplitude)
_amplitude = _maxAmplitude;

_initialAmplitude = _amplitude;
_time = 0;
_totalTime = time;
_speed = speed;
_direction = direction;
}

public function update( time:Number ):void
{
if (_totalTime == 0)
return;

_wave += _speed / _maxAmplitude;
_amplitude
```

```
= _initialAmplitude - _initialAmplitude / _totalTime * _time;

if (_time < _totalTime)
{

switch (_direction)
{
case 1:
_game3D.x = Math.cos(_wave) * _amplitude;
break;

case 2:
_game3D.y = Math.sin(_wave / 1.5) * _amplitude;
break;

case 3:
_game3D.x = Math.cos(_wave) * _amplitude;
_game3D.y = Math.sin(_wave / 1.5) * _amplitude;
break;
}
} else
{
_game3D.x = _game3D.y = 0;
clearValues();
}

_time += time;
}

public function clearLayer( layer:String ):void
{
var viewLayer:*;
switch (layer)
{
case ViewEnum.ALERT:
case ViewEnum.ERROR:
case ViewEnum.HOVER:
case ViewEnum.MODAL:
case ViewEnum.UI:
break;
case ViewEnum.GAME:
if (Application.STARLING_ENABLED)
viewLayer = _game3D;
else
viewLayer = _game;
break;
case ViewEnum.BACKGROUND_LAYER:
if (Application.STARLING_ENABLED)
viewLayer = _background3DLayer;
else
viewLayer
```

```
= _backgroundLayer;
break;
case ViewEnum.GAME_LAYER:
if (Application.STARLING_ENABLED)
viewLayer = _game3DLayer;
else
viewLayer = _gameLayer;
break;
}
while (viewLayer.numChildren > 0)
viewLayer.removeChildAt(0);
}

public function clearValues():void
{
_wave = _amplitude = _time = _speed = _direction = _totalTime = 0;
}

private function addStage3DLayers( e:* ):void
{
Application.STARLING_ENABLED = true;
//Starling.context.driverInfo.toLowerCase().indexOf("software") == -1;
if (Application.STARLING_ENABLED)
{
_away3D = new View3D();
_away3D.stage3DProxy = _stage3DProxy;
_away3D.shareContext = true;

//3d camera
//_away3D.camera.position = new Vector3D(DeviceMetrics.WIDTH_PIXELS * -.5,
DeviceMetrics.HEIGHT_PIXELS * -.5, 800);
//_away3D.camera.lookAt(new Vector3D(DeviceMetrics.WIDTH_PIXELS * -.5,
DeviceMetrics.HEIGHT_PIXELS * -.5, 0));

//orthogonal camera
_away3D.camera.position = new Vector3D(DeviceMetrics.WIDTH_PIXELS * -.5, 0, 1300);
_away3D.camera.lens = new OrthographicLens(1300);
_away3D.camera.lookAt(new Vector3D(DeviceMetrics.WIDTH_PIXELS * -.5,
DeviceMetrics.HEIGHT_PIXELS * -.5, 0));

var skyLight:DirectionalLight = new DirectionalLight(0, -1, 0);
skyLight.diffuse = .9;
skyLight.specular = 0.1;
skyLight.ambientColor = 0xffffff;
_away3D.scene.addChild(skyLight);

_contextView.addChildAt(_away3D, 0);
lightPicker = new StaticLightPicker([skyLight]);

Parsers.enableAllBundled();

_starling
```

```
= new Starling(org.starling.display.Sprite, _contextView.stage, _stage3DProxy.viewPort,
_stage3DProxy.stage3D); //new Rectangle(0, 0, DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS)); //, null, "auto", Context3DProfile.BASELINE); //, null,
"software");
//_starling.addEventListener(org.starling.events.Event.ROOT_CREATED, addStage3DLayers);
_starling.start();

_stage3DProxy.addEventListener(flash.events.Event.ENTER_FRAME, onEnterFrame);

_starling.stage.addChild(_game3D);
_background3DLayer = new org.starling.display.Sprite();
_background3DLayer.touchable = false;
_game3DLayer = new org.starling.display.Sprite();
_game3DLayer.touchable = false;
_game3D.addChild(_background3DLayer);
_game3D.addChild(_game3DLayer);
_game3D.alpha = 0.999;
_starling.addEventListener(org.starling.events.Event.CONTEXT3D_CREATE,
onContextCreated);
//if (CONFIG::DEBUG == true)
//_starling.showStatsAt("center", "bottom");
} else
{
_starling.stop();
_starling.dispose();
_starling = null;
_contextView.addChildAt(_game, 0);
_backgroundLayer = new flash.display.Sprite();
_backgroundLayer.mouseChildren = _backgroundLayer.mouseEnabled = false;
_gameLayer = new flash.display.Sprite();
_gameLayer.mouseChildren = _gameLayer.mouseEnabled = false;
_game.addChild(_backgroundLayer);
_game.addChild(_gameLayer);
}
logSpecs();
}

private function onContextCreated( e:org.starling.events.Event ):void
{
_eventDispatcher.dispatchEvent(new StateEvent(StateEvent.LOST_CONTEXT));
}

private static function logSpecs():void
{
// Get the playerâ€™s version by using the flash.system.Capabilities class.
var versionNumber:String = Capabilities.version;
var log:String = "[SPECS] CPU:" + Capabilities.cpuArchitecture + " -- OS:" + Capabilities.os;
var versionArray:Array = versionNumber.split(",");
var platformAndVersion:Array = versionArray[0].split(" ");

log
```

```
+= " -- FLASH_PLAYER:" + (int(platformAndVersion[1]) + "." + int(versionArray[1]));
log += " -- RESOLUTION:" + Capabilities.screenResolutionX + "x" +
Capabilities.screenResolutionY;
log += " -- GPU:" + (Application.STARLING_ENABLED ? "true" : "false");
TimeLog.addLog(log);
}

private function onEnterFrame( event:flash.events.Event ):void
{
// Render the Starling animation layer
_starling.nextFrame();

// Render the Away3D layer
_away3D.render();
}

[Inject]
public function set contextView( value:flash.display.DisplayObjectContainer ):void {
_contextView = value; }
[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
[Inject]
public function set frameTickProvider( value:ITickProvider ):void { _frameTickProvider = value; }
}
}

------------------------------------
File 672: igw\com\ui\core\component\IComponent.as
package com.ui.core.component
{
public interface IComponent
{
function get enabled():Boolean;
function set enabled( value:Boolean ):void;

function destroy():void;
}
}

------------------------------------
File 673: igw\com\ui\core\component\accordian\AccordianButton.as
package com.ui.core.component.accordian
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.button.ButtonLabelFormat;

import flash.display.Bitmap;
import
```

```
flash.display.Sprite;
import flash.events.MouseEvent;

import org.osflash.signals.Signal;

public class AccordianButton extends Sprite
{
public static const NORMAL:int = 0;
public static const LOCKED:int = 1;
public static const NOT_COMPLETED:int = 2;
public static const COMPLETED:int = 3;

private var _actionSignal:Signal;
private var _button:BitmapButton;
private var _checkbox:BitmapButton;
private var _data:*;
private var _groupID:String;
private var _height:Number;
private var _id:String;
private var _lock:Bitmap;
private var _state:int;

public function init( groupID:String, id:String, title:String, state:int, actionSignal:Signal,
buttonType:String, width:Number, height:Number, data:* = null ):void
{
_actionSignal = actionSignal;
_button = UIFactory.getButton(buttonType, width, height, 0, 0, title, id == null ?
LabelEnum.TITLE : LabelEnum.SUBTITLE);
_button.selectable = true;
addEventListener(MouseEvent.CLICK, onClick, false, 0, true);
addChild(_button);

_data = data;
_groupID = groupID;
_height = height;
_id = id;

this.state = state;
}

private function onClick( e:MouseEvent ):void
{
if (!_lock)
_actionSignal.dispatch(_groupID, _id, _data);
}

public function set labelFormat( v:ButtonLabelFormat ):void { _button.labelFormat = v; }

public function get id():String { return _id; }

override
```

```
public function get height():Number { return _height; }

public function get selected():Boolean { return _button.selected; }
public function set selected( v:Boolean ):void { _button.selected = v; }

public function set state( v:int ):void
{
if (_checkbox && contains(_checkbox))
removeChild(_checkbox);
if (_lock && contains(_lock))
removeChild(_lock);
_state = v;
switch (_state)
{
case NORMAL:
_checkbox = UIFactory.destroyButton(_checkbox);
_lock = UIFactory.destroyPanel(_lock);
_button.enabled = true;
break;
case LOCKED:
_checkbox = UIFactory.destroyButton(_checkbox);
if (!_lock)
_lock = UIFactory.getBitmap("IconLockMiniBMD");
_lock.x = 18;
_lock.y = 4.5;
addChild(_lock);
_button.enabled = false;
break;
case NOT_COMPLETED:
_checkbox = UIFactory.destroyButton(_checkbox);
if (!_checkbox)
_checkbox = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 18, 4);
_checkbox.enabled = true;
_checkbox.enabled = _checkbox.selected = false;
addChild(_checkbox);
_lock = UIFactory.destroyPanel(_lock);
_button.enabled = true;
break;
case COMPLETED:
_checkbox = UIFactory.destroyButton(_checkbox);
if (!_checkbox)
_checkbox = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 18, 4);
_checkbox.enabled = true;
_checkbox.selected = true;
_checkbox.enabled = false;
addChild(_checkbox);
_lock = UIFactory.destroyPanel(_lock);
_button.enabled = true;
break;
}
}
```

```
public function get text():String { if (_button) return _button.text; else return ''; }
public function set text( v:String ):void { if (_button) _button.text = v; }

public function destroy():void
{
x = y = 0;
state = NORMAL;
_actionSignal = null;
removeEventListener(MouseEvent.CLICK, onClick);
_button = UIFactory.destroyButton(_button);
_data = null;
}
}
}
```

File 674: igw\com\ui\core\component\accordian\AccordianComponent.as

```
package com.ui.core.component.accordian
{
import com.ui.core.component.IComponent;
import com.ui.core.component.button.ButtonLabelFormat;

import flash.display.Sprite;
import flash.events.Event;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class AccordianComponent extends Sprite implements IComponent
{
private var _actionSignal:Signal;
private var _enabled:Boolean;
private var _height:Number;
private var _groupLookup:Dictionary;
private var _groups:Vector.<AccordianGroup>;
private var _tempGroup:AccordianGroup;
private var _width:Number;

public function init( width:Number, height:Number ):void
{
_actionSignal = new Signal(String, String, Object);
_actionSignal.add(onAction);
enabled = true;
_groupLookup = new Dictionary(true);
_groups = new Vector.<AccordianGroup>;
_height = height;
_width = width;
}
```

```
public function addGroup( id:String, title:String ):void
{
if (!_groupLookup.hasOwnProperty(id))
{
_tempGroup = ObjectPool.get(AccordianGroup);
_tempGroup.init(id, title, _width, _height, _actionSignal);
addChild(_tempGroup);
_groupLookup[id] = _tempGroup;
_groups.push(_tempGroup);
layout();
}
}

public function getGroup( id:String ):AccordianGroup
{
if (_groupLookup.hasOwnProperty(id))
return _groupLookup[id];
return null;
}

public function setGroupTitle( id:String, title:String, labelFormat:ButtonLabelFormat = null ):void
{
if (_groupLookup.hasOwnProperty(id))
{
_tempGroup = _groupLookup[id];
_tempGroup.headerButton.text = title;
if (labelFormat)
_tempGroup.headerButton.labelFormat = labelFormat;
}
}

public function setSelected( groupID:String, subItemID:String ):void
{
onAction(groupID, subItemID, null);
}

public function setSelectedSubItemByIndex( groupID:String, index:int ):void
{
if (_groupLookup.hasOwnProperty(groupID))
{
_tempGroup = _groupLookup[groupID];
if (_tempGroup.hasSubItems && _tempGroup.subItems.length > index)
setSelected(groupID, _tempGroup.subItems[index].id);
}
}

public function addSubItemToGroup( groupID:String, subItemID:String, title:String, state:int
):void
{
```

```
if (_groupLookup.hasOwnProperty(groupID))
{
_tempGroup = _groupLookup[groupID];
_tempGroup.addSubItem(subItemID, title, state);
layout();
}
}

public function setSubItemState( groupID:String, subItemID:String, state:int ):void
{
if (_groupLookup.hasOwnProperty(groupID))
{
_tempGroup = _groupLookup[groupID];
_tempGroup.setSubItemState(subItemID, state);
layout();
}
}

public function removeSubItemFromGroup( groupID:String, subItemID:String ):void
{
if (_groupLookup.hasOwnProperty(groupID))
{
_tempGroup = _groupLookup[groupID];
_tempGroup.removeSubItem(subItemID);
layout();
}
}

public function removeGroup( id:String ):void
{
if (_groupLookup.hasOwnProperty(id))
{
_tempGroup = _groupLookup[id];
delete _groupLookup[id];
_groups.splice(_groups.indexOf(_tempGroup), 1);
removeChild(_tempGroup);
ObjectPool.give(_tempGroup);
layout();
}
}

private function onAction( groupID:String, subItemID:String, data:* ):void
{
for (var i:int = 0; i < _groups.length; i++)
{
_tempGroup = _groups[i];
_tempGroup.headerButton.selected = _tempGroup.id == groupID;
_tempGroup.expanded = _tempGroup.id == groupID;
if (_tempGroup.subItems && (subItemID || groupID != _tempGroup.id))
{
```

```
for (var j:int = 0; j < _tempGroup.subItems.length; j++)
{
_tempGroup.subItems[j].selected = _tempGroup.subItems[j].id == subItemID && _tempGroup.id
== groupID;
}
}
}
removeEventListener(Event.ENTER_FRAME, update);
addEventListener(Event.ENTER_FRAME, update, false, 0, true);
}

private function update( e:Event ):void
{
var active:Boolean = false;
for (var i:int = 0; i < _groups.length; i++)
{
if (_groups[i].update())
active = true;
}
layout();
if (!active)
removeEventListener(Event.ENTER_FRAME, update);
}

private function layout():void
{
var ypos:Number = 0;
for (var i:int = 0; i < _groups.length; i++)
{
_groups[i].y = ypos;
ypos += _groups[i].height + 2;
}
}

public function addListener( listener:Function ):void { _actionSignal.add(listener); }
public function removeListener( listener:Function ):void { _actionSignal.remove(listener); }

public function get enabled():Boolean { return false; }
public function set enabled( value:Boolean ):void { _enabled = value; }

public function destroy():void
{
x = y = 0;
_actionSignal.removeAll();
_actionSignal = null;
removeEventListener(Event.ENTER_FRAME, update);
for (var id:String in _groupLookup)
removeGroup(id);
_groupLookup = null;
_groups.length
```

```
= 0;
_groups = null;
_tempGroup = null;
}
}
}

------------------------------------
File 675: igw\com\ui\core\component\accordian\AccordianGroup.as
package com.ui.core.component.accordian
{
import com.enum.ui.ButtonEnum;

import flash.display.Sprite;
import flash.geom.Rectangle;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class AccordianGroup extends Sprite
{
private var _actionSignal:Signal;
private var _expanded:Boolean;
private var _headerButton:AccordianButton;
private var _height:Number;
private var _id:String;
private var _maxHeight:Number;
private var _subItemHolder:Sprite;
private var _subItemLookup:Dictionary;
private var _subItemRect:Rectangle;
private var _subItems:Vector.<AccordianButton>;
private var _tempSubItem:AccordianButton;
private var _width:Number;

public function init( id:String, title:String, width:Number, height:Number, actionSignal:Signal
):void
{
_actionSignal = actionSignal;
_expanded = false;
_headerButton = ObjectPool.get(AccordianButton);
_headerButton.init(id, null, title, 0, _actionSignal, ButtonEnum.HEADER, width, height);
addChild(_headerButton);

_id = id;
_subItemLookup = new Dictionary(true);
_subItemRect = new Rectangle(0, 0, width, 0);
_subItemHolder = new Sprite();
_subItemHolder.scrollRect = _subItemRect;
_subItemHolder.y = _headerButton.height;
addChild(_subItemHolder);
```

```
_subItems = new Vector.<AccordianButton>;
_height = height;
_width = width;
}

public function addSubItem( id:String, title:String, state:int = 0 ):void
{
if (!_subItemLookup.hasOwnProperty(id))
{
_tempSubItem = ObjectPool.get(AccordianButton);
_tempSubItem.init(_id, id, title, state, _actionSignal, ButtonEnum.ACCORDIAN_SUBITEM,
_width, 28);
_subItemLookup[id] = _tempSubItem;
_subItemHolder.addChild(_tempSubItem);
_subItems.push(_tempSubItem);
layout();
}
}

public function setSubItemState( id:String, state:int ):void
{
if (_subItemLookup.hasOwnProperty(id))
{
_tempSubItem = _subItemLookup[id];
_tempSubItem.state = state;
layout();
}
}

public function removeSubItem( id:String ):void
{
if (_subItemLookup.hasOwnProperty(id))
{
_tempSubItem = _subItemLookup[id];
delete _subItemLookup[id];
_subItems.splice(_subItems.indexOf(_tempSubItem), 1);
_subItemHolder.removeChild(_tempSubItem);
ObjectPool.give(_tempSubItem);
_tempSubItem = null;
layout();
}
}

public function update():Boolean
{
if (_expanded)
{
if (_subItemRect.height < _maxHeight)
{
_subItemRect.height
```

```
+= 15;
if (_subItemRect.height >= _maxHeight)
{
_subItemRect.height = _maxHeight;
_subItemHolder.scrollRect = _subItemRect;
return false;
}
}
} else
{
if (_subItemRect.height > 0)
{
_subItemRect.height -= 15;
if (_subItemRect.height <= 0)
{
_subItemRect.height = 0;
_subItemHolder.scrollRect = _subItemRect;
return false;
}
}
}
_subItemHolder.scrollRect = _subItemRect;
return true;
}

public function getSubItem( id:String ):AccordianButton
{
if (_subItemLookup.hasOwnProperty(id))
return _subItemLookup[id];
return null;
}

private function layout():void
{
var ypos:Number = 0;
for (var i:int = 0; i < _subItems.length; i++)
{
_subItems[i].y = ypos;
ypos += _subItems[i].height;
}
_maxHeight = ypos;
}

public function get expanded():Boolean { return _expanded; }
public function set expanded( v:Boolean ):void { _expanded = v; }

public function get hasSubItems():Boolean { return _subItems && _subItems.length > 1; }
public function get headerButton():AccordianButton { return _headerButton; }

override public function get height():Number { return _headerButton.height +
_subItemRect.height
```

```
- 1; }

public function get id():String { return _id; }

public function get subItems():Vector.<AccordianButton> { return _subItems; }
public function get text():String { return _headerButton.text; }

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);
_actionSignal = null;
ObjectPool.give(_headerButton);
_headerButton = null;
for (var i:String in _subItemLookup)
removeSubItem(i);
_subItems.length = 0;
_subItems = null;
_subItemLookup = null;
_subItemRect = null;
_subItemHolder = null;
_tempSubItem = null;
}
}
}

------------------------------------
File 676: igw\com\ui\core\component\bar\ProgressBar.as
package com.ui.core.component.bar
{
import com.ui.core.component.IComponent;

import flash.display.DisplayObject;
import flash.display.Sprite;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.greensock.TweenLite;
import org.greensock.easing.Quad;

public class ProgressBar extends Sprite implements IComponent
{
public static const HORIZONTAL:int = 0;
public static const VERTICAL:int = 1;

private var _amount:Number;
private var _base:DisplayObject;
private var _dwidth:Number;
private var _dheight:Number;
private var _min:Number;
private
```

```
var _max:Number;
private var _orientation:int;
private var _overlayHolder:Sprite;
private var _overlay:DisplayObject;
private var _tweenThreshold:Number;
private var _mask:DisplayObject;
private var _reverse:Boolean;
private var _tweenSpeed:Number = .5;

protected const _logger:ILogger = getLogger('ProgressBar');

public function init( orientation:int, overlay:DisplayObject = null, base:DisplayObject = null,
tweenThreshold:Number = 0.15, reverse:Boolean = false ):void
{
_overlayHolder = new Sprite();
_amount = _dwidth = _dheight = _min = _max = 0;
_orientation = orientation;
_base = base;
_overlay = overlay;
_tweenThreshold = tweenThreshold;
_reverse = reverse;

if (_base)
addChild(_base);

if (_overlay)
{
_dwidth = _overlay.width;
_dheight = _overlay.height;

if (_orientation == VERTICAL)
{
_overlay.scaleY = -1;
_overlayHolder.y = _overlay.height;
}

_overlayHolder.addChild(_overlay);
addChild(_overlayHolder);

if (_base)
{
_overlayHolder.x = (_base.width - _overlay.width) / 2;
_overlayHolder.y = (_base.height - _overlay.height) / 2;
}
}
}

public function setMinMax( min:Number, max:Number ):void
{
_min = min;
_max
```

```
= max;
amount = _amount;
}

public function createDefaultOverlay( color:uint, dalpha:Number, dwidth:Number,
dheight:Number ):void
{
amount = _amount;
}

public function setBase( base:DisplayObject ):void
{
if (_base)
removeChild(_base);

_base = base;

if (_base)
{
addChildAt(_base, 0);

if (_overlayHolder)
{
_overlayHolder.x = (_base.width - _overlay.width) / 2;
_overlayHolder.y = (_base.height - _overlay.height) / 2;
}
}
}

public function setOverlay( overlay:DisplayObject ):void
{
if (_overlay)
_overlayHolder.removeChild(_overlay);

_overlay = overlay;
_dwidth = _overlay.width;
_dheight = _overlay.height;

if (_orientation == VERTICAL)
{
var holder:Sprite = new Sprite();
_overlay.scaleY = -1;
_overlayHolder.y = _overlay.height;
}

_overlayHolder.addChild(_overlay);
}

public function addMask( mask:DisplayObject ):void
{
_mask
```

```
= mask;
mask.cacheAsBitmap = true;
_overlayHolder.mask = mask;

if (_base)
{
mask.x = (_base.width - _overlay.width) / 2;
mask.y = (_base.height - _overlay.height) / 2;
}

addChild(mask);
_overlayHolder.cacheAsBitmap = true;
}

public function setSize( w:Number, h:Number ):void
{
if (_overlay)
{
_overlay.width = w;
_overlay.height = h;
}

if (_base)
{
_base.width = w;
_base.height = h;
}
}

public function get amount():Number { return _amount; }
public function set amount( v:Number ):void
{
var oldAmount:Number = _amount;
_amount = v;

if (_amount < _min)
_amount = _min;

if (_amount > _max)
_amount = _max;

var oldScale:Number = (oldAmount - _min) / (_max - _min);
var newScale:Number = (_amount - _min) / (_max - _min);
var nwidth:Number = (_orientation == HORIZONTAL) ? _dwidth * newScale : _dwidth;
var nheight:Number = (_orientation == HORIZONTAL) ? _dheight : _dheight * newScale;

if (!_overlayHolder)
return;

if (Math.abs(newScale - oldScale) > _tweenThreshold)
{
```

```
if (_reverse)
TweenLite.to(_overlayHolder, _tweenSpeed, {width:nwidth, height:nheight, x:(_dwidth - nwidth),
y:(_dheight - nheight), ease:Quad.easeIn});
else
TweenLite.to(_overlayHolder, _tweenSpeed, {width:nwidth, height:nheight, ease:Quad.easeIn});
} else
{
TweenLite.killTweensOf(_overlayHolder);

_overlayHolder.width = nwidth;
_overlayHolder.height = nheight;

if (_reverse)
{
if (_orientation == HORIZONTAL)
{
_overlayHolder.x = _dwidth - nwidth;
} else
{
_overlayHolder.y = _dheight - nheight;
}
}
}
}

public function get base():DisplayObject { return _base; }
public function get overlay():DisplayObject { return _overlay; }

public function set overrideAmount( v:Number ):void
{
var threshold:Number = _tweenThreshold;
_tweenThreshold = 100000;
amount = v;
_tweenThreshold = threshold;
}

public function get enabled():Boolean { return visible; }
public function set enabled( value:Boolean ):void { visible = value; }

public function get barWidth():Number { return _overlayHolder.width; }

public function set tweenSpeed( v:Number ):void { _tweenSpeed = v; }

public function destroy():void
{
filters = [];

while (numChildren > 0)
removeChildAt(0);

if
```

```
(_overlay && _overlay.parent)
_overlay.parent.removeChild(_overlay);

_overlayHolder = null;
_base = null;
_overlay = null;
_tweenSpeed = .5;
visible = true;
}
}
}
```

File 677: igw\com\ui\core\component\bar\Slider.as

```
package com.ui.core.component.bar
{
import com.Application;
import com.enum.ui.ButtonEnum;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.core.ScaleBitmap;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.DisplayObject;
import flash.display.Sprite;
import flash.display.Stage;
import flash.events.MouseEvent;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;

public class Slider extends Sprite
{
public var onSliderUpdate:Signal;

private var _selector:BitmapButton;
private var _bg:ScaleBitmap;
private var _bar:Bitmap;
private var _currentText:Label;
private var _minText:Label;
private var _maxText:Label;

private var _minValue:Number;
private var _maxValue:Number;
private
```

```
var _tick:Number;
private var _currentValue:Number;
private var _percent:Number;

private var _width:Number;
private var _yOffset:Number;

public function Slider()
{
super();

onSliderUpdate = new Signal(Number, Number);

_bg = UIFactory.getScaleBitmap(PanelEnum.INPUT_BOX_BLUE);

_bar = UIFactory.getBitmap(PanelEnum.SCROLL_BAR);

_selector = UIFactory.getButton(ButtonEnum.SLIDER);
_selector.useHandCursor = true;

_currentText = new Label(14, 0xffffff, 100, 20, false, 1);
_currentText.align = TextFormatAlign.CENTER;

_minText = new Label(16, 0xffffff, 100, 20, false, 1);
_minText.align = TextFormatAlign.LEFT;

_maxText = new Label(16, 0xffffff, 100, 20, false, 1);
_maxText.align = TextFormatAlign.RIGHT;

addChild(_bg);
addChild(_bar);
addChild(_selector);
addChild(_currentText);
addChild(_minText);
addChild(_maxText);
}

public function init( width:Number, height:Number, min:Number, max:Number,
parent:DisplayObject ):void
{
_minValue = min;
_maxValue = max;
_tick = _maxValue / 10;

_bg.width = width;
_bg.height = height;

_width = _bg.width - _selector.width * 0.5;

_selector.addEventListener(MouseEvent.MOUSE_DOWN, onSelectorDown, false, 0, true);
_selector.addEventListener(MouseEvent.MOUSE_UP,
```

```
onSelectorUp, false, 0, true);

if (parent)
{
parent.addEventListener(MouseEvent.MOUSE_UP, onSelectorUp, false, 0, true);
parent.addEventListener(MouseEvent.ROLL_OUT, onSelectorUp, false, 0, true);
}

layout();

_minText.text = String(min);
_maxText.text = String(max);
}

private function layout():void
{
_bar.height = _bg.height - 6;

_selector.x = _bg.x
_selector.y = _bg.y - 3;

_bar.x = _bg.x + 4;
_bar.y = _bg.y + 3;

_currentText.x = _bg.x + (_bg.width - _currentText.width) * 0.5;

_maxText.x = _bg.x + _bg.width - _maxText.width;

_minText.y = _maxText.y = _currentText.y = _bg.y + _bg.height + 5;
}

public function set enabled( enabled:Boolean ):void
{
if (enabled)
{
_selector.visible = true;
_selector.addEventListener(MouseEvent.MOUSE_DOWN, onSelectorDown, false, 0, true);
_selector.addEventListener(MouseEvent.MOUSE_UP, onSelectorUp, false, 0, true);

} else
{
_selector.visible = false;
_selector.removeEventListener(MouseEvent.MOUSE_DOWN, onSelectorDown);
_selector.removeEventListener(MouseEvent.MOUSE_UP, onSelectorUp);
}
}

public function set currentValue( currentValue:Number ):void
{
_currentValue = clamp(currentValue);
updateSelector();
```

```
}

public function get currentValue():Number
{
return _currentValue;
}

private function updateSelector():void
{
_percent = ((_currentValue - _minValue) / (_maxValue - _minValue));
_selector.x = _percent * _width;
_bar.width = _selector.x - _bar.x + _selector.width * 0.5;

_currentText.text = String(_currentValue);
}

private function onSelectorDown( e:MouseEvent ):void
{
_yOffset = mouseX - _selector.x;
_selector.addEventListener(MouseEvent.MOUSE_MOVE, onSelectorMove, false, 0, true);

if (parent)
{
parent.addEventListener(MouseEvent.MOUSE_UP, onSelectorUp, false, 0, true);
parent.addEventListener(MouseEvent.ROLL_OUT, onSelectorUp, false, 0, true);
parent.addEventListener(MouseEvent.MOUSE_MOVE, onSelectorMove, false, 0, true);
}
}

private function onSelectorMove( e:MouseEvent ):void
{
var x:int = mouseX - _yOffset;
_selector.x = x;

if (_selector.x > _width)
_selector.x = _width;

if (_selector.x < _bg.x)
_selector.x = _bg.x;

var range:Number = _maxValue - _minValue;
_percent = Math.floor(_selector.x / _width * 100);
_currentValue = range * _percent / 100 + _minValue;

_bar.width = _selector.x - _bar.x + _selector.width * 0.5;

onSliderUpdate.dispatch(_currentValue, _percent);
_currentText.text = String(_currentValue);
}

private
```

```
function onSelectorUp( e:MouseEvent ):void
{
_yOffset = mouseX - _selector.x;
_selector.removeEventListener(MouseEvent.MOUSE_MOVE, onSelectorMove);

if (parent)
{
parent.removeEventListener(MouseEvent.MOUSE_UP, onSelectorUp);
parent.removeEventListener(MouseEvent.ROLL_OUT, onSelectorUp);
parent.removeEventListener(MouseEvent.MOUSE_MOVE, onSelectorMove);
}
}

public function destroy():void
{

_selector.removeEventListener(MouseEvent.MOUSE_DOWN, onSelectorDown);
_selector.removeEventListener(MouseEvent.MOUSE_UP, onSelectorUp);
_currentText.destroy();
}


private function clamp( val:Number ):Number
{
return Math.max(_minValue, Math.min(_maxValue, val))
}

}
}

-----------------------------------
File 678: igw\com\ui\core\component\bar\VScrollbar.as
package com.ui.core.component.bar
{
import com.ui.core.ScaleBitmap;
import com.ui.core.component.IComponent;

import flash.display.BitmapData;
import flash.display.DisplayObject;
import flash.display.DisplayObjectContainer;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;

public class VScrollbar extends Sprite implements IComponent
{
private
```

```
static const DRAG_THRESHOLD:int = 15;
private static const MIN_DRAGGER_SIZE:int = 17;
private static const PADDING:int = 4;

private var _dragBar:Sprite;
private var _dragBarBG:ScaleBitmap;
private var _scrollBG:Sprite;
private var _parent:DisplayObjectContainer;

private var _offsetY:int;
private var _top:int;
private var _bottom:int;
private var _dragging:Boolean;
private var _position:Number;
private var _scrollRange:int;
private var _maxDraggerSize:int;

private var _totalVisibleHeight:Number;
private var _totalScrollableHeight:Number;
private var _scrollPercent:Number;
private var _scrollableHeightRatio:Number;

private var _dragBarUp:BitmapData
private var _dragBarRollOver:BitmapData

private var _scrollableObject:DisplayObject;

private var _startFullyScrolled:Boolean;

private var _isEnabled:Boolean;
private var _alreadyMovedThisFrame:Boolean;

private var _lastScroll:Number;

private var _maxScroll:Number = 0;
private var _minScroll:Number = 0;
private var _totalDelta:Number = 0;
private var _delta:Number = 0;

// This signal is dispatched when scrolling occurs.
public var onScrollSignal:Signal;

public function init( width:Number, height:Number, xPos:Number, yPos:Number,
dragBarRect:Rectangle, dragBarMCName:String = 'DragBarBGMC', dragBarName:String =
'DragBarUpBMD', dragBarRollOverName:String =
'DragBarRollOverBMD', startFullyScrolled:Boolean = false, parent:DisplayObjectContainer =
null, scrollableObject:DisplayObject = null ):void
{
x = xPos;
y = yPos;
_dragging
```

```
= false;
_parent = parent;
_position = 0;
if (dragBarMCName != '')
{
var dragBarBGClass:Class = Class(getDefinitionByName(dragBarMCName));
_scrollBG = Sprite(new dragBarBGClass());
_scrollBG.x = 0;
_scrollBG.y = 0;
_scrollBG.height = height;
_scrollBG.width = width;
} else
{
_scrollBG = new Sprite();
_scrollBG.alpha = 0;
_scrollBG.graphics.drawRect(0, 0, width, height);
}
addChild(_scrollBG);

if (dragBarRollOverName != '')
{
var dragBarRollOverBMD:Class = Class(getDefinitionByName(dragBarRollOverName));
_dragBarRollOver = BitmapData(new dragBarRollOverBMD());
}

var dragBarUpBMD:Class = Class(getDefinitionByName(dragBarName));
_dragBarUp = BitmapData(new dragBarUpBMD());

_dragBarBG = new ScaleBitmap(_dragBarUp);
_dragBarBG.scale9Grid = dragBarRect;
_dragBarBG.smoothing = true;

_dragBar = new Sprite();
_dragBar.x = 0 + 3;
_dragBar.y = 0 + _scrollBG.height - _dragBar.height;
_dragBar.addChild(_dragBarBG);
addChild(_dragBar);

_top = _scrollBG.y + 2;
_maxDraggerSize = height - PADDING;
_scrollRange = height - _dragBar.height - PADDING;
_bottom = _top + _scrollRange;

_dragBarBG.width = width;

_maxScroll = height;

onScrollSignal = new Signal(Number);

_scrollableObject = scrollableObject;

_totalScrollableHeight
```

```
= 0;

_startFullyScrolled = startFullyScrolled;

if (_startFullyScrolled)
_scrollPercent = 1;
else
_scrollPercent = 0;

if (_scrollRange > 0)
addListeners();
}

public function updateScrollbarHeight( newHeight:Number ):void
{
_scrollBG.height = newHeight;
_maxDraggerSize = _scrollBG.height - PADDING;
}

public function updateDisplayedHeight( newHeight:Number ):void
{
if (!isNaN(newHeight))
{
_totalVisibleHeight = newHeight;
updateScrollableHeightRatio();
}
}

public function updateScrollableHeight( updatedScrollableHeight:Number ):void
{
if (!isNaN(updatedScrollableHeight))
{
_totalScrollableHeight = updatedScrollableHeight;
updateScrollableHeightRatio();
}
}

private function updateScrollableHeightRatio():void
{
_scrollableHeightRatio = _totalScrollableHeight / _totalVisibleHeight;
if (_scrollableHeightRatio <= 1 || isNaN(_scrollableHeightRatio))
{
_scrollableHeightRatio = 1;
enabled = false;
} else
{
if (enabled == false)
enabled = true;
}

_dragBarBG.height
```

```
= Math.max(_maxDraggerSize / _scrollableHeightRatio, MIN_DRAGGER_SIZE);

_scrollRange = _scrollBG.height - _dragBar.height - PADDING;

if (_scrollRange > 0)
addListeners();
else
removeListeners();

_bottom = _top + _scrollRange;
var y:Number = _scrollPercent * _scrollRange + _top;
y = y < _top ? _top : y > _bottom ? _bottom : y;
_dragBar.y = y;
}

public function updateScrollY( y:Number ):void
{
y = y < _top ? _top : y > _bottom ? _bottom : y;
_scrollPercent = (_scrollRange > 0) ? (y - _top) / _scrollRange : 0;
_dragBar.y = y;

onScrollSignal.dispatch(_scrollPercent);
}

public function updateScrollPercent( percent:Number ):void
{
_scrollPercent = percent;
_scrollPercent = _scrollPercent > 1 ? 1 : _scrollPercent < 0 ? 0 : _scrollPercent;
var y:Number = _scrollPercent * _scrollRange + _top;
y = y < _top ? _top : y > _bottom ? _bottom : y;
_dragBar.y = y;

onScrollSignal.dispatch(_scrollPercent);
}

private function addListeners():void
{
_dragBar.useHandCursor = true;
_dragBar.buttonMode = true;

_scrollBG.buttonMode = true;
_scrollBG.useHandCursor = true;

_dragBar.addEventListener(MouseEvent.MOUSE_DOWN, onMouseDown, false, 0, true);
_scrollBG.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);

if (_scrollableObject)
_scrollableObject.addEventListener(MouseEvent.MOUSE_WHEEL, handleMouseWheel, false,
0, true);
else if (_parent)
{
```

```
_parent.addEventListener(MouseEvent.CLICK, onParentClick, false, 0, true);
_parent.addEventListener(MouseEvent.MOUSE_DOWN, onDragScrollAreaDown);
_parent.addEventListener(MouseEvent.MOUSE_WHEEL, handleMouseWheel, false, 0, true);
}
}

private function removeListeners():void
{
_dragBar.useHandCursor = false;
_dragBar.buttonMode = false;

_scrollBG.buttonMode = false;
_scrollBG.useHandCursor = false;

_scrollBG.removeEventListener(MouseEvent.CLICK, onMouseClick);
_dragBar.removeEventListener(MouseEvent.MOUSE_DOWN, onMouseDown);
if (_parent)
{
_parent.removeEventListener(MouseEvent.CLICK, onParentClick);
_parent.removeEventListener(MouseEvent.ROLL_OUT, onMouseUp);
_parent.removeEventListener(MouseEvent.MOUSE_DOWN, onDragScrollAreaDown);
_parent.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
_parent.removeEventListener(MouseEvent.MOUSE_UP, onMouseUp);
}

if (_scrollableObject)
_scrollableObject.removeEventListener(MouseEvent.MOUSE_WHEEL, handleMouseWheel);
else if (_parent)
_parent.removeEventListener(MouseEvent.MOUSE_WHEEL, handleMouseWheel);
}

private function onMouseDown( event:MouseEvent ):void
{
_offsetY = mouseY - _dragBar.y;
_dragging = true;
if (_parent)
{
_parent.addEventListener(MouseEvent.ROLL_OUT, onMouseUp, false, 0, true);
_parent.addEventListener(MouseEvent.MOUSE_MOVE, onMouseMove, false, 0, true);
_parent.addEventListener(MouseEvent.MOUSE_UP, onMouseUp, false, 0, true);
}
event.stopPropagation();
}

private function onDragScrollAreaDown( event:MouseEvent ):void
{
_offsetY = mouseY;
_position = _dragBar.y;
if (_parent)
{
```

```
_parent.addEventListener(MouseEvent.ROLL_OUT, onMouseUp, false, 0, true);
_parent.addEventListener(MouseEvent.MOUSE_MOVE, onMouseDrag, false, 0, true);
_parent.addEventListener(MouseEvent.MOUSE_UP, onMouseUp, false, 0, true);
}
event.stopPropagation();
}

private function onMouseDrag( event:MouseEvent ):void
{
var y:Number = _offsetY - mouseY + _position;
if (_dragging || Math.abs(y) > DRAG_THRESHOLD)
{
_dragging = true;
updateScrollY(y);
}
event.stopPropagation();
}

private function onMouseMove( event:MouseEvent ):void
{
var y:Number = mouseY - _offsetY;
updateScrollY(y);
event.stopPropagation();
}

private function onMouseUp( event:MouseEvent ):void
{
_offsetY = mouseY - _dragBar.y;
_dragging = false;
if (_parent)
{
_parent.removeEventListener(MouseEvent.ROLL_OUT, onMouseUp);
_parent.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
_parent.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseDrag);
_parent.removeEventListener(MouseEvent.MOUSE_UP, onMouseUp);
}
event.stopPropagation();
}

private function onMouseClick( event:MouseEvent ):void
{
updateScrollY(mouseY);
event.stopPropagation();
}

private function onParentClick( event:MouseEvent ):void
{
if (_dragging)
event.stopPropagation();
}
```

```
public function handleMouseWheel( event:MouseEvent ):void
{
if (!_alreadyMovedThisFrame)
{
_alreadyMovedThisFrame = true;
addEventListener(Event.EXIT_FRAME, onExitFrame, false, 0, true);
_delta = event.delta;
}
_totalDelta += event.delta;
event.stopImmediatePropagation();
event.stopPropagation();
}

private function onExitFrame( e:Event ):void
{
removeEventListener(Event.EXIT_FRAME, onExitFrame);
_alreadyMovedThisFrame = false;

var scrollCount:Number = Math.abs(_totalDelta / _delta);
if (_totalDelta > 0)
scrollUp(scrollCount);
else
scrollDown(scrollCount);

_totalDelta = 0;
}

public function set enabled( value:Boolean ):void
{
_isEnabled = value;
_dragBar.visible = _isEnabled;
_scrollBG.visible = _isEnabled;
if (_isEnabled && _scrollRange > 0)
addListeners();
else
removeListeners();
}

public function scrollUp( scrollCount:Number = 1 ):void
{
var percent:Number = (_maxScroll * scrollCount) / (_totalScrollableHeight - _totalVisibleHeight);
updateScrollY(_dragBar.y - _scrollRange * percent);
}

public function scrollDown( scrollCount:Number = 1 ):void
{
var percent:Number = (_maxScroll * scrollCount) / (_totalScrollableHeight - _totalVisibleHeight);
updateScrollY(_dragBar.y + (_scrollRange * percent));
}
```

```
public function resetScroll():void
{
if (_startFullyScrolled)
{
_scrollPercent = 1;
_dragBar.y = _bottom;
} else
{
_scrollPercent = 0;
_dragBar.y = _top;
}

onScrollSignal.dispatch(_scrollPercent);
}

public function get enabled():Boolean { return _isEnabled; }

public function set maxScroll( v:Number ):void { _maxScroll = v; }
public function set minScroll( v:Number ):void { _minScroll = v; }

public function get percent():Number { return _scrollPercent; }

public function destroy():void
{
removeListeners();
_dragBar = null;
_parent = null;
_scrollableObject = null;
_scrollBG = null;
onScrollSignal.removeAll();
onScrollSignal = null;
}
}
}

-----------------------------------
File 679: igw\com\ui\core\component\button\BitmapButton.as
ï»¿package com.ui.core.component.button
{
import com.enum.ui.ButtonEnum;
import com.ui.core.ScaleBitmap;

import flash.display.BitmapData;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.osflash.signals.Signal;

public
```

```
class BitmapButton extends ButtonBase
{
protected var _bitmap:ScaleBitmap = new ScaleBitmap();
private var _upSkin:BitmapData;
private var _overSkin:BitmapData;
private var _downSkin:BitmapData;
private var _disabledSkin:BitmapData;
private var _selectSkin:BitmapData;

public var onStateChange:Signal;

public function init( upSkin:BitmapData, overSkin:BitmapData = null, downSkin:BitmapData =
null, disabledSkin:BitmapData = null, selectSkin:BitmapData = null ):void
{
_upSkin = upSkin;
_overSkin = overSkin;
_downSkin = downSkin;
_disabledSkin = disabledSkin;
_selectSkin = selectSkin;

addChild(_bitmap);

_state = ButtonEnum.STATE_NORMAL;
_selectable = _selected = false;
horzTxtMargin = vertTxtMargin = DEFAULT_MARGIN;
addListeners();
enabled = true;

onStateChange = new Signal(BitmapButton);
}

public function updateBackgrounds( upSkin:BitmapData, overSkin:BitmapData = null,
downSkin:BitmapData = null, disabledSkin:BitmapData = null, selectSkin:BitmapData = null
):void
{
if (upSkin)
_upSkin = upSkin;

if (overSkin)
_overSkin = overSkin;

if (downSkin)
_downSkin = downSkin;

if (disabledSkin)
_disabledSkin = disabledSkin;

if (selectSkin)
_selectSkin = selectSkin;

showState();
```

```
}

override protected function showState():void
{
super.showState();
switch (_state)
{
case ButtonEnum.STATE_NORMAL:
if (_upSkin)
_bitmap.bitmapData = _upSkin;
else
_bitmap.bitmapData = null;
if (_labelFormat && _label)
{
_label.bold = _labelFormat.upBold;
_label.textColor = _labelFormat.upColor;
_label.text = _label.text;
}
break;
case ButtonEnum.STATE_OVER:
if (_overSkin)
_bitmap.bitmapData = _overSkin;
if (_labelFormat && _label)
{
_label.bold = _labelFormat.roBold;
_label.textColor = _labelFormat.roColor;
_label.text = _label.text;
}
break;
case ButtonEnum.STATE_DOWN:
if (_downSkin)
_bitmap.bitmapData = _downSkin;
if (_labelFormat && _label)
{
_label.bold = _labelFormat.downBold;
_label.textColor = _labelFormat.downColor;
_label.text = _label.text;
}
break;
case ButtonEnum.STATE_SELECTED:
if (_selectSkin)
_bitmap.bitmapData = _selectSkin;
if (_labelFormat && _label)
{
_label.bold = _labelFormat.selectedBold;
_label.textColor = _labelFormat.selectedColor;
_label.text = _label.text;
}
break;
case
```

```
ButtonEnum.STATE_DISABLED:
if (_disabledSkin)
_bitmap.bitmapData = _disabledSkin;
else
{
if (_upSkin)
_bitmap.bitmapData = _upSkin;
}
if (_labelFormat && _label)
{
_label.bold = _labelFormat.disabledBold;
_label.textColor = _labelFormat.disabledColor;
_label.text = _label.text;
}
break;
}
_bitmap.smoothing = true;
}

private function stateChanged( event:Event ):void
{
if (onStateChange)
onStateChange.dispatch(this);
}

override protected function onMouse( e:MouseEvent ):void
{
var oldState:String = _state;
super.onMouse(e);
if (_state != oldState)
{
stateChanged(e);
}
}

override public function setSize( width:int, height:int ):void
{
if (width == 0 && _upSkin)
width = _upSkin.width;
if (height == 0 && _upSkin)
height = _upSkin.height;
_bitmap.setSize(width, height);
setLabelSize(width, height);
}

override public function get defaultSkinHeight():Number { return _bitmap.height; }
override public function get defaultSkinWidth():Number { return _bitmap.width; }

override public function set height( value:Number ):void { _bitmap.height = value; }
override public function set width( value:Number ):void { _bitmap.width = value; }

override
```

```
public function set scale9Grid( innerRectangle:Rectangle ):void { _bitmap.scale9Grid =
innerRectangle; }

override public function destroy():void
{
super.destroy();
if (_bitmap)
{
setSize(0, 0);
_bitmap.bitmapData = null;
_bitmap.visible = true;
}
_upSkin = null;
_overSkin = null;
_downSkin = null;
_disabledSkin = null;
_selectSkin = null;
onStateChange.removeAll();
}
}
}

-----------------------------------
File 680: igw\com\ui\core\component\button\ButtonBase.as
package com.ui.core.component.button
{
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.IComponent;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.events.MouseEvent;

public class ButtonBase extends Sprite implements IComponent
{
protected static const DEFAULT_MARGIN:int = 3;

protected var _leftMargin:Number = 0;
protected var _rightMargin:Number = 0;
protected var _bottomMargin:Number = 0;
protected var _topMargin:Number = 0;

protected var _font:int = -1;
protected var _label:Label;
protected var _labelFormat:ButtonLabelFormat;
protected var _labelType:String;
protected var _selected:Boolean = false;
protected
```

```
var _selectable:Boolean = false;
protected var _state:String;
protected var _onRollOverSound:String = 'sounds/sfx/AFX_UI_Mouse2_V001A.mp3';
protected var _onClickSound:String = 'sounds/sfx/AFX_UI_Mouse1_V001A.mp3';

protected function onMouse( e:MouseEvent ):void
{
if (mouseEnabled)
{
switch (e.type)
{
case MouseEvent.MOUSE_DOWN:
_state = ButtonEnum.STATE_DOWN;
break;
case MouseEvent.MOUSE_UP:
_state = _selected ? ButtonEnum.STATE_SELECTED : ButtonEnum.STATE_OVER;
if (_onClickSound != '')
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_1, 0.5);
break;
case MouseEvent.ROLL_OVER:
_state = (_selected) ? ButtonEnum.STATE_SELECTED : ButtonEnum.STATE_OVER;
if (_onRollOverSound != '')
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_2, 0.5);
break;
case MouseEvent.ROLL_OUT:
_state = (_selected) ? ButtonEnum.STATE_SELECTED : ButtonEnum.STATE_NORMAL;
break;
case MouseEvent.CLICK:
if (_selectable)
{
_selected = !_selected;
_state = _selected ? ButtonEnum.STATE_SELECTED : ButtonEnum.STATE_OVER;
}
break;
}
showState();
}
//e.stopPropagation();
}

protected function showState():void {}

public function setMargin( horzMargin:Number, vertMargin:Number ):void
{
if (_label)
{
if (!isNaN(vertMargin))
vertTxtMargin = vertMargin;

if (!isNaN(horzMargin))
horzTxtMargin
```

```
= horzMargin;

resizeAndLayoutLabel();
}
}

protected function resizeAndLayoutLabel():void
{
_label.setSize(defaultSkinWidth - (_leftMargin + _rightMargin), defaultSkinHeight - (_topMargin
+ _bottomMargin));
_label.x = _leftMargin + (defaultSkinWidth - _leftMargin - _rightMargin - _label.width) / 2;
_label.y = _topMargin + (defaultSkinHeight - _topMargin - _bottomMargin - (_label.height -
(_label.height - _label.textHeight) / 2)) / 2;
}

public function setSize( width:int, height:int ):void {}
public function setLabelSize( width:int, height:int ):void { if (_label) _label.setSize(width, height);
}

protected function addListeners():void
{
addEventListener(MouseEvent.CLICK, onMouse, false, 0, true);
addEventListener(MouseEvent.MOUSE_UP, onMouse, false, 0, true);
addEventListener(MouseEvent.MOUSE_DOWN, onMouse, false, 0, true);
addEventListener(MouseEvent.ROLL_OVER, onMouse, false, 0, true);
addEventListener(MouseEvent.ROLL_OUT, onMouse, false, 0, true);
}

protected function removeListeners():void
{
removeEventListener(MouseEvent.CLICK, onMouse);
removeEventListener(MouseEvent.MOUSE_UP, onMouse);
removeEventListener(MouseEvent.MOUSE_DOWN, onMouse);
removeEventListener(MouseEvent.ROLL_OVER, onMouse);
removeEventListener(MouseEvent.ROLL_OUT, onMouse);
}

public function get defaultSkinWidth():Number { throw new Error('ButtonBase:[defaultSkinWidth],
must be overriden by subclass.'); return null; }
public function get defaultSkinHeight():Number { throw new
Error('ButtonBase:[defaultSkinHeight], must be overriden by subclass.'); return null; }

public function get enabled():Boolean { return mouseEnabled; }
public function set enabled( value:Boolean ):void
{
mouseEnabled = value;
buttonMode = value;
_state = value ? ((_selected) ? ButtonEnum.STATE_SELECTED :
ButtonEnum.STATE_NORMAL) : (_selected) ? ButtonEnum.STATE_SELECTED :
ButtonEnum.STATE_DISABLED;
showState();
```

```
}

public function set font( value:int ):void { _font = value; }
public function set fontSize( fontSize:int ):void
{
if (_label)
{
_label.fontSize = fontSize;
resizeAndLayoutLabel();
}
}

public function get label():Label { return _label; }
public function set labelFormat( v:ButtonLabelFormat ):void { _labelFormat = v; showState(); }
public function set labelType( v:String ):void { _labelType = v; }

public function get selectable():Boolean { return _selectable; }
public function set selectable( value:Boolean ):void { _selectable = value; }

public function get selected():Boolean { return _selected; }
public function set selected( value:Boolean ):void
{
_selected = value;
_state = _selected ? ButtonEnum.STATE_SELECTED : ButtonEnum.STATE_NORMAL;
if (enabled)
showState();
}

public function set state( value:String ):void { _state = value; showState(); }

public function get text():String { return (_label) ? _label.text : ""; }
public function set text( msg:String ):void
{
if (!_label)
{
if (_labelType)
_label = UIFactory.getLabel(_labelType, 50, 50, 0, 0);
else if (_font>=0)
_label = new Label(50, 0, 50, 50, true, _font);
else
_label = new Label(50, 0, 50, 50);
_label.constrictTextToSize = true;
addChild(_label);
}
_label.text = msg;
resizeAndLayoutLabel();
showState();
}

public
```

```
function set textColor( color:uint ):void { if (_label) _label.textColor = color; }

public function get textColor():uint { return _label.textColor; }


public function set horzTxtMargin( horzMargin:Number ):void
{
if (_label && !isNaN(horzMargin))
{
_leftMargin = _rightMargin = horzMargin;
resizeAndLayoutLabel();
}
}

public function set vertTxtMargin( vertMargin:Number ):void
{
if (_label && !isNaN(vertMargin))
{
_topMargin = _bottomMargin = vertMargin;
resizeAndLayoutLabel();
}
}

public function destroy():void
{
hitArea = null;
_labelFormat = null;
_labelType = null;
while (numChildren > 0)
removeChildAt(0);
removeListeners();
_label = UIFactory.destroyLabel(_label);
x = y = 0;
alpha = scaleX = scaleY = 1;
visible = true;
}
}
}
```

File 681: igw\com\ui\core\component\button\ButtonLabelFormat.as

```
package com.ui.core.component.button
{
public class ButtonLabelFormat
{
private var _disabledBold:Boolean;
private var _disabledColor:uint;

private var _downBold:Boolean;
private var _downColor:uint;

private
```

```
var _roBold:Boolean;
private var _roColor:uint;

private var _selectedBold:Boolean;
private var _selectedColor:uint;

private var _upBold:Boolean;
private var _upColor:uint;

public function ButtonLabelFormat( data:Object )
{
_upBold = data.upBold;
_upColor = data.upColor;

var a:Array = ["disabled", "down", "ro", "selected"];
for (var i:int = 0; i < a.length; i++)
{
this["_" + a[i] + "Bold"] = (data[a[i] + "Bold"]) != null ? (data[a[i] + "Bold"]) : _upBold;
this["_" + a[i] + "Color"] = (data[a[i] + "Color"]) != null ? (data[a[i] + "Color"]) : _upColor;
}
}

public function get disabledBold():Boolean { return _disabledBold; }
public function get disabledColor():uint { return _disabledColor; }

public function get downBold():Boolean { return _downBold; }
public function get downColor():uint { return _downColor; }

public function get roBold():Boolean { return _roBold; }
public function get roColor():uint { return _roColor; }

public function get selectedBold():Boolean { return _selectedBold; }
public function get selectedColor():uint { return _selectedColor; }

public function get upBold():Boolean { return _upBold; }
public function get upColor():uint { return _upColor; }
}
}
```

File 682: igw\com\ui\core\component\button\ButtonLabelFormatFactory.as

```
package com.ui.core.component.button
{
import com.enum.ui.ButtonEnum;

import flash.utils.Dictionary;

public class ButtonLabelFormatFactory
{
private static const _lookup:Dictionary = new Dictionary();

public
```

```
static function getFormat( type:String ):ButtonLabelFormat
{
if (_lookup.hasOwnProperty(type))
return _lookup[type];

var format:Object;
var labelFormat:ButtonLabelFormat;

switch (type)
{
case ButtonEnum.BLUE_A:
format = {upBold:true, upColor:0xd1e5f7,
roBold:true, roColor:0xddffff,
downBold:true, downColor:0x48637a,
disabledBold:true, disabledColor:0x48637a};
break;
case ButtonEnum.GOLD_A:
format = {upBold:true, upColor:0xffe3b1,
roBold:true, roColor:0xfff2dd,
downBold:true, downColor:0x7a6948,
disabledBold:true, disabledColor:0x7a6948};
break;
case ButtonEnum.RED_A:
format = {upBold:true, upColor:0xffacac,
roBold:true, roColor:0xffddde,
downBold:true, downColor:0x7a4948,
disabledBold:true, disabledColor:0x7a4948};
break;
case ButtonEnum.GREEN_A:
format = {upBold:true, upColor:0xacffb4,
roBold:true, roColor:0xddffe0,
downBold:true, downColor:0x487a4f,
disabledBold:true, disabledColor:0x487a4f};
break;
case ButtonEnum.ACCORDIAN_SUBITEM:
format = {upBold:false, upColor:0x213745,
roBold:false, roColor:0xd1e5f7,
downBold:false, downColor:0x213745,
selectedBold:true, selectedColor:0xd1e5f7};
break;
case ButtonEnum.HEADER:
case ButtonEnum.HEADER_NOTCHED:
format = {upBold:false, upColor:0x213745,
roBold:false, roColor:0x213745,
downBold:false, downColor:0x213745,
selectedBold:true, selectedColor:0xd1e5f7};
break;
}

if (format)
{
```

```
labelFormat = new ButtonLabelFormat(format);
_lookup[type] = labelFormat;
}

return labelFormat;
}

}
}

-----------------------------------
File 683: igw\com\ui\core\component\contextmenu\ContextMenu.as
ï»¿package com.ui.core.component.contextmenu
{
import com.Application;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.IComponent;
import com.ui.core.component.label.Label;
import com.ui.core.effects.ViewEffects;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.utils.Dictionary;

import org.parade.enum.PlatformEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class ContextMenu extends View implements IComponent
{
private var _selections:Vector.<IContextMenuItem>;
private var _multiChoiceLookup:Dictionary;
private var _bg:ScaleBitmap;
private var _title:Label;
private var _subText:Label;
private var _width:Number;
private var _height:Number;
private var _isEnabled:Boolean;
private var _clickedXPos:int;
private var _clickedYPos:int;
private var _boundsXPos:int;
private var _boundsYPos:int;

private
```

```
var _choiceYPos:Number;

public function ContextMenu()
{
_selections = new Vector.<IContextMenuItem>();
_bg = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER_DARK);
_bg.width = width;
_bg.height = height;

_title = UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, width, 25, _bg.x, 1);
_title.multiline = true;
_title.constrictTextToSize = false;
_title.autoSize = TextFieldAutoSize.CENTER;
_title.textColor = 0xacd1ff;
_title.leading = -5;

_subText = UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, width, 25, _title.x, _title.y -
3);
_subText.fontSize = 12;
_subText.autoSize = TextFieldAutoSize.CENTER;
_subText.constrictTextToSize = false;
}

public function setup( contextMenuTitle:String, xPos:Number = 0, yPos:Number = 0,
width:Number = 150, boundsX:int = 0, boundsY:int = 0, contextMenuSubText:String = " ):void
{
if (DeviceMetrics.PLATFORM == PlatformEnum.MOBILE){
//scaleX = scaleY = DeviceMetrics.DENSITY >= 1 ? DeviceMetrics.DENSITY + .1 : 1;

//scaleX = scaleY = 4;
}

if(CONFIG::IS_MOBILE){
scaleX = scaleY = 2;
}


destroyOnRollout = true;

clearSelections();
_bg.width = width;
_width = width;

_clickedXPos = xPos;
_clickedYPos = yPos;

_boundsXPos = boundsX;
_boundsYPos = boundsY;

_title.width = width - 6;
_title.x
```

```
= (width - _title.width) * .5;

if (contextMenuTitle != null)
_title.text = contextMenuTitle;

_subText.x = (width - _subText.width) * .5;
_subText.y = _title.height - 3;
_subText.text = contextMenuSubText;

_height = _subText.y + _title.height + _subText.textHeight;
_bg.height = _height;

_choiceYPos = _subText.y + _subText.textHeight + 8;

addChild(_bg);
addChild(_title);
addChild(_subText);
}

public function addContextMenuChoice( displayName:String, callback:Function, args:Array,
isEnabled:Boolean = true, tooltipText:String = '', color:uint = 0xffffff ):void
{
var newChoice:ContextMenuItem = new ContextMenuItem(displayName, callback, args,
isEnabled, tooltipText, color);
newChoice.onSelectionClicked.add(onSelectionClick);
newChoice.x = _bg.x + (_bg.width - newChoice.width) * 0.5;
newChoice.y = _choiceYPos;
_choiceYPos = newChoice.y + newChoice.height;

addChild(newChoice);
_selections.push(newChoice);

layout();
}

public function addContextMenuMultiChoice( category:String, startingIndex:uint,
indexChanged:Function = null, heldShift:uint = 26, heldTime:Number = 1500 ):void
{
if (!(category in _multiChoiceLookup))
{
var newChoice:ContextMenuMultiChoiceItem = new ContextMenuMultiChoiceItem(category,
startingIndex, indexChanged, heldShift, heldTime);
newChoice.onSelectionClicked.add(onSelectionClick);

if (indexChanged != null)
newChoice.onSelectedIndexUpdated.add(onMultiChoiceIndexUpdate);

newChoice.x = _bg.x + (_bg.width - newChoice.width) * 0.5;
newChoice.y = _choiceYPos;
_choiceYPos = newChoice.y + newChoice.height;

addChild(newChoice);
```

```
_selections.push(newChoice);
_multiChoiceLookup[category] = newChoice;
layout();
}
}

public function addChoiceToMultiChoice( category:String, displayName:String,
callback:Function, args:Array, isEnabled:Boolean, tooltip:String, color:uint = 0xffffff ):void
{
if (category in _multiChoiceLookup)
_multiChoiceLookup[category].addChoiceItem(displayName, callback, args, isEnabled, tooltip,
color);
}

private function onMultiChoiceIndexUpdate( callback:Function, index:int ):void
{
if (callback != null && _presenter != null)
callback.apply(_presenter, [index]);
}

private function layout():void
{
var len:uint = _selections.length;
var lastChoice:Sprite = Sprite(_selections[len - 1]);
_height = lastChoice.y + lastChoice.height;
_bg.height = _height + 15;

var newXPos:int = _clickedXPos - _width * 0.5;
var newYPos:int = _clickedYPos - _height * 0.5;

if (newYPos + _height > _boundsYPos)
{
newYPos -= (newYPos + _height) - _boundsYPos;
} else if (newYPos + _height < 0)
{
newYPos = 0;
}

if (newXPos + _width > _boundsXPos)
{
newXPos -= (newXPos + _width) - _boundsXPos;
} else if (newXPos < 0)
{
newXPos = 0;
}

x = newXPos;
y = newYPos;
}

private
```

```
function onSelectionClick( callback:Function, args:Array ):void
{
callback.apply(this, args);
_viewFactory.destroyView(this);
ObjectPool.give(this);
}

private function onRollOut( e:MouseEvent ):void
{
_viewFactory.destroyView(this);
ObjectPool.give(this);
}

private function clearSelections():void
{
var len:uint = _selections.length;
for (var i:uint = 0; i < len; ++i)
{
_selections[i].destroy();
removeChild(Sprite(_selections[i]));
_selections[i] = null;
}
_selections.length = 0;

_multiChoiceLookup = new Dictionary();
}

public function set destroyOnRollout( v:Boolean ):void
{
removeListener(this, MouseEvent.ROLL_OUT, onRollOut);
if (v)
{
addListener(this, MouseEvent.ROLL_OUT, onRollOut);
addListener(Application.STAGE, MouseEvent.MOUSE_DOWN, onStageMouseDown);
}
}

private function onStageMouseDown( e:MouseEvent ):void
{
if (!hitTestPoint(e.stageX, e.stageY, false))
onRollOut(e);
}

public function get enabled():Boolean { return false; }
public function set enabled( value:Boolean ):void { _isEnabled = value; }

override public function get height():Number { return _height; }
override public function get width():Number { return _width; }

override public function get bounds():Rectangle { return getBounds(parent); }
override
```

```
public function get effects():ViewEffects { return null; }
public function get selections():Vector.<IContextMenuItem> { return _selections; }
override public function get typeUnique():Boolean { return false; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }

override public function destroy():void
{
clearSelections();
super.destroy();
if (parent)
parent.removeChild(this);
}
}
}

-----------------------------------
File 684: igw\com\ui\core\component\contextmenu\ContextMenuItem.as
package com.ui.core.component.contextmenu
{
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.modal.ButtonFactory;

import flash.display.Sprite;
import flash.events.MouseEvent;

import org.osflash.signals.Signal;

public class ContextMenuItem extends Sprite implements IContextMenuItem
{
public var onSelectionClicked:Signal;

private var _btn:BitmapButton;
private var _data:ContextMenuItemData;

public function ContextMenuItem( displayName:String, callback:Function, args:Array,
isEnabled:Boolean, tooltip:String, color:uint = 0xffffff )
{
onSelectionClicked = new Signal(Function, Array);

_data = new ContextMenuItemData(displayName, callback, args, isEnabled, tooltip, color);

_btn = UIFactory.getButton(ButtonEnum.BLUE_A, 129, 25, 0, 0, displayName);
_btn.labelFormat = null;
_btn.textColor = color;
_btn.enabled = isEnabled;
_btn.addEventListener(MouseEvent.MOUSE_DOWN, onKillMouseDown, false, 1, true);
_btn.addEventListener(MouseEvent.CLICK,
```

```
onBtnClick, false, 1, true);
addChild(_btn);
}

public function get tooltip():String { return _data.tooltip; }

private function onBtnClick( e:MouseEvent ):void
{
e.stopPropagation();
if (_data.callback != null)
onSelectionClicked.dispatch(_data.callback, _data.args);
}

private function onKillMouseDown( e:MouseEvent ):void
{
e.stopPropagation();
}

public function destroy():void
{
onSelectionClicked.removeAll();
onSelectionClicked = null;

if (_btn)
{
_btn.removeEventListener(MouseEvent.MOUSE_DOWN, onKillMouseDown);
_btn.removeEventListener(MouseEvent.MOUSE_UP, onBtnClick);
_btn.destroy();
}
_btn = null;


_data = null;
}
}
}

-----------------------------------
File 685: igw\com\ui\core\component\contextmenu\ContextMenuItemData.as
package com.ui.core.component.contextmenu
{
public class ContextMenuItemData
{
private var _displayName:String
private var _callback:Function;
private var _args:Array;
private var _isEnabled:Boolean;
private var _tooltip:String
private var _color:uint;

public
```

```
function ContextMenuItemData( displayName:String, callback:Function, args:Array,
isEnabled:Boolean, tooltip:String, color:uint = 0xffffff )
{
_displayName = displayName;
_callback = callback;
_args = args;
_isEnabled = isEnabled;
_tooltip = tooltip;
_color = color;
}

public function get displayName():String { return _displayName; }
public function get callback():Function { return _callback; }
public function get args():Array { return _args; }
public function get isEnabled():Boolean { return _isEnabled; }
public function get tooltip():String { return _tooltip; }
public function get color():uint { return _color; }
}
}

-----------------------------------
File 686: igw\com\ui\core\component\contextmenu\ContextMenuMultiChoiceItem.as
package com.ui.core.component.contextmenu
{
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.utils.Timer;

import org.osflash.signals.Signal;

public class ContextMenuMultiChoiceItem extends Sprite implements IContextMenuItem
{
public var onSelectionClicked:Signal;
public var onSelectedIndexUpdated:Signal;

private var _btn:BitmapButton;
private var _arrowLeft:BitmapButton;
private var _arrowRight:BitmapButton;

private var _data:Vector.<ContextMenuItemData>;

private var _heldShift:uint;

private var _selectedIndex:int;
private var _heldCount:int;

private
```

```
var _heldTime:Number;

private var _positive:Boolean;

private var _indexChanged:Function;

private var _timer:Timer;

public function ContextMenuMultiChoiceItem( category:String, startingIndex:uint,
indexChanged:Function, heldShift:uint, heldTime:Number )
{
onSelectionClicked = new Signal(Function, Array);
onSelectedIndexUpdated = new Signal(Function, int);

_data = new Vector.<ContextMenuItemData>;

_selectedIndex = startingIndex;
_heldShift = heldShift;
_heldTime = heldTime;
_indexChanged = indexChanged;

_timer = new Timer(heldTime);
_timer.addEventListener(TimerEvent.TIMER, onTimerTick, false, 0, true);

_arrowLeft = UIFactory.getButton(ButtonEnum.BACK_ARROW);
_arrowLeft.addEventListener(MouseEvent.MOUSE_DOWN, onShiftLeft, false, 0, true);
_arrowLeft.addEventListener(MouseEvent.CLICK, onEndTimer, false, 0, true);

_arrowRight = UIFactory.getButton(ButtonEnum.FORWARD_ARROW);
_arrowRight.addEventListener(MouseEvent.MOUSE_DOWN, onShiftRight, false, 0, true);
_arrowRight.addEventListener(MouseEvent.CLICK, onEndTimer, false, 0, true);

_btn = UIFactory.getButton(ButtonEnum.BLUE_A, 129, 25);
_btn.labelFormat = null;
_btn.addEventListener(MouseEvent.MOUSE_DOWN, onKillMouseDown, false, 1, true);
_btn.addEventListener(MouseEvent.CLICK, onBtnClick, false, 1, true);

_btn.x = _arrowLeft.width + 3;

_arrowLeft.y = _btn.y + (_btn.height - _arrowLeft.height) * 0.5;
_arrowLeft.visible = false;

_arrowRight.x = _btn.x + _btn.width + 3;
_arrowRight.y = _btn.y + (_btn.height - _arrowRight.height) * 0.5;
_arrowRight.visible = false;

addChild(_btn);
addChild(_arrowLeft);
addChild(_arrowRight);
}

public
```

```
function addChoiceItem( displayName:String, callback:Function, args:Array, isEnabled:Boolean,
tooltip:String, color:uint = 0xffffff ):void
{
var data:ContextMenuItemData = new ContextMenuItemData(displayName, callback, args,
isEnabled, tooltip, color)
_data.push(data);

if ((_data.length - 1) == _selectedIndex && _btn != null)
{
_btn.text = displayName;
_btn.textColor = color;
_btn.enabled = isEnabled;
}

if (!_arrowRight.visible && _data.length > 1)
_arrowRight.visible = true;

if (!_arrowLeft.visible && _data.length > 1)
_arrowLeft.visible = true;
}

public function get tooltip():String { return (_selectedIndex < _data.length) ?
_data[_selectedIndex].tooltip : ''; }


private function onBtnClick( e:MouseEvent ):void
{
if ((_selectedIndex < _data.length))
{
var data:ContextMenuItemData = _data[_selectedIndex];
e.stopPropagation();
if (data.callback != null)
onSelectionClicked.dispatch(data.callback, data.args);
}
}

private function onShiftLeft( e:MouseEvent ):void
{
updateSelectedIndex(_selectedIndex - 1);

_positive = false;
_heldCount = Math.floor(_selectedIndex / _heldShift) - 1;
_timer.start();
}

private function onShiftRight( e:MouseEvent ):void
{
updateSelectedIndex(_selectedIndex + 1);

_positive = true;
_heldCount
```

```
= Math.floor(_selectedIndex / _heldShift) + 1;
_timer.start();
}

private function onTimerTick( e:TimerEvent ):void
{
var len:uint = _data.length;
var maxCount:uint = len / _heldShift;

if (_heldCount > maxCount)
_heldCount = 0;
else if (_heldCount < 0)
_heldCount = maxCount;

updateSelectedIndex(_heldShift * _heldCount)

if (_positive)
++_heldCount;
else
--_heldCount;
}

private function onEndTimer( e:MouseEvent ):void
{
_heldCount = 0;
_timer.stop();
_timer.reset();
}

private function onKillMouseDown( e:MouseEvent ):void
{
e.stopPropagation();
}

private function updateSelectedIndex( index:uint ):void
{
_selectedIndex = index;
var max:uint = _data.length - 1;
if (_selectedIndex > max)
_selectedIndex = 0;
else if (_selectedIndex < 0)
_selectedIndex = max;

_btn.text = _data[_selectedIndex].displayName;

if (_indexChanged != null)
onSelectedIndexUpdated.dispatch(_indexChanged, _selectedIndex);
}

public function destroy():void
{
```

```
if (onSelectionClicked)
onSelectionClicked.removeAll();

onSelectionClicked = null;

if (onSelectedIndexUpdated)
onSelectedIndexUpdated.removeAll();

onSelectedIndexUpdated = null;

if (_timer)
{
if (_timer.running)
_timer.stop();

_timer.removeEventListener(TimerEvent.TIMER, onTimerTick);
}

_timer = null;

if (_btn)
{
_btn.removeEventListener(MouseEvent.MOUSE_DOWN, onKillMouseDown);
_btn.removeEventListener(MouseEvent.MOUSE_UP, onBtnClick);
_btn.destroy();
}
_btn = null;

if (_arrowLeft)
{
_arrowLeft.removeEventListener(MouseEvent.MOUSE_UP, onShiftLeft);
_arrowLeft.removeEventListener(MouseEvent.CLICK, onEndTimer);
_arrowLeft.destroy();
}
_arrowLeft = null;

if (_arrowRight)
{
_arrowRight.removeEventListener(MouseEvent.MOUSE_UP, onShiftRight);
_arrowRight.removeEventListener(MouseEvent.CLICK, onEndTimer);
_arrowRight.destroy();
}
_arrowRight = null;

_data.length = 0;
}
}
}

-----------------------------------
```

File 687: igw\com\ui\core\component\contextmenu\IContextMenuItem.as

```
package com.ui.core.component.contextmenu
{
public interface IContextMenuItem
{
function get tooltip():String;
function destroy():void;
}
}
```

File 688: igw\com\ui\core\component\filterlist\FilterButton.as

```
package com.ui.core.component.filterlist
{
import com.ui.core.component.button.BitmapButton;

import flash.display.Sprite;

public class FilterButton extends Sprite
{
private var _btn:BitmapButton;
private var _filter:*;
private var _index:int;
private var _padding:Number;

public function FilterButton( filter:*, index:int, btn:BitmapButton, padding:Number )
{
_filter = filter
_index = index;
_btn = btn;
_padding = padding;
_btn.selectable = true;

addChild(_btn);
}

override public function get height():Number
{
return _btn.height;
}

override public function get width():Number
{
return _btn.width;
}

public function get filter():* { return _filter; }
public function get index():int { return _index; }
public function get padding():Number { return _padding; }

public
```

```
function set selected( value:Boolean ):void { _btn.selected = value; }
public function get selected():Boolean { return _btn.selected; }

public function set enabled( value:Boolean ):void { _btn.enabled = value; }
public function get enabled():Boolean { return _btn.enabled; }

public function destroy():void
{
_btn.destroy();
}
}
}
```

File 689: igw\com\ui\core\component\filterlist\FilterHeader.as

```
package com.ui.core.component.filterlist
{
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

public class FilterHeader extends Sprite
{
private var _text:Label;
private var _bg:Bitmap;
private var _index:int;
private var _headerPadding:Number;
private var _padding:Number;
private var _filterBtns:Vector.<FilterButton>;

public function FilterHeader( text:Label, bg:String, index:int, padding:Number,
headerPadding:Number)
{
super();
var filterHeaderBG:Class = Class(getDefinitionByName((bg)));

_filterBtns = new Vector.<FilterButton>;

_text = text;
_index = index;
_headerPadding = headerPadding;
_padding = padding;

_bg = new Bitmap(BitmapData(new filterHeaderBG()));

_text.x = _bg.x + (_bg.width - _text.width) * 0.5;
_text.y
```

```
= _bg.y + (_bg.height - _text.height) * 0.5;

addChild(_bg);
addChild(_text);
}

public function addFilter( filterBtn:FilterButton ):void
{
_filterBtns.push(filterBtn);
}

public function sortFilters( sort:Function ):void
{
_filterBtns.sort(sort);
}

public function get index():int { return _index; }
public function get filterBtns():Vector.<FilterButton> { return _filterBtns; }
public function get filterCount():uint { return _filterBtns.length; }
public function get padding():Number { return _padding; }
public function get headerPadding():Number { return _headerPadding; }
}
}
```

File 690: igw\com\ui\core\component\filterlist\FilterList.as

```
package com.ui.core.component.filterlist
{
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.ui.Mouse;

import org.adobe.utils.StringUtil;
import org.osflash.signals.Signal;

public class FilterList extends Sprite implements IComponent
{
private var _padding:Number;
private var _filterBtns:Vector.<FilterButton>
private var _selectedFilterBtns:Vector.<FilterButton>;
private var _filter:Array;

private var _onlyOneFilterSelected:Boolean;
private var _unselectOnReclick:Boolean;
private var _isHorizontal:Boolean;

public var onSelectionChanged:Signal;

public
```

```
function FilterList()
{
_filterBtns = new Vector.<FilterButton>;
_selectedFilterBtns = new Vector.<FilterButton>;
_filter = new Array;
onSelectionChanged = new Signal(Array);
super();
}

public function init( bg:String = '', onlyOneFilterSelected:Boolean = true,
unselectOnReclick:Boolean = true, isHorizontal:Boolean = true ):void
{
_onlyOneFilterSelected = onlyOneFilterSelected;
_unselectOnReclick = unselectOnReclick;
_isHorizontal = isHorizontal;
}

public function addFilterBtn( filter:*, btn:BitmapButton, padding:Number = 0, index:int = -1 ):void
{
if (index == -1)
index = _filterBtns.length;

var newFilterBtn:FilterButton = new FilterButton(filter, index, btn, padding);
newFilterBtn.addEventListener(MouseEvent.CLICK, onFilterClicked, false, 0, true);
addChild(newFilterBtn);
_filterBtns.push(newFilterBtn);
_filterBtns.sort(orderItems);

layout();
}

public function selectFilterByIndex( index:int ):void
{
if (index < _filterBtns.length)
{
selectFilterBtn(_filterBtns[index]);
_filterBtns[index].selected = true;
}
}

public function selectFilterByFilter( v:Array ):void
{
if (v)
{
var len:uint = _filterBtns.length;
var currentFilterBtn:FilterButton;
for (var i:uint = 0; i < len; ++i)
{
currentFilterBtn = _filterBtns[i];
if (v.toString().toLowerCase() == currentFilterBtn.filter.toString().toLowerCase())
{
```

```
selectFilterBtn(_filterBtns[i]);
_filterBtns[i].selected = true;
}
}
}
}

private function orderItems( itemOne:FilterButton, itemTwo:FilterButton ):int
{
var sortOrderOne:Number = itemOne.index;
var sortOrderTwo:Number = itemTwo.index;

if (sortOrderOne < sortOrderTwo)
{
return -1;
} else if (sortOrderOne > sortOrderTwo)
{
return 1;
} else
{
return 0;
}
}

protected function layout():void
{
var len:uint = _filterBtns.length;
var currentFilterBtn:FilterButton;
var xPos:Number = 0;
var yPos:Number = 0;
for (var i:uint = 0; i < len; ++i)
{
currentFilterBtn = _filterBtns[i];
currentFilterBtn.x = xPos;
currentFilterBtn.y = yPos;

if (_isHorizontal)
xPos += currentFilterBtn.width + currentFilterBtn.padding;
else
yPos += currentFilterBtn.height + currentFilterBtn.padding;
}
}

private function selectFilterBtn( filterBtn:FilterButton ):void
{
var index:int = _selectedFilterBtns.indexOf(filterBtn);

if (_onlyOneFilterSelected && _selectedFilterBtns.length > 0)
{
if
```

```
(index == -1 && _unselectOnReclick == false)
clearSelections();
}

if (index == -1)
{
_selectedFilterBtns.push(filterBtn);
_filter.push(filterBtn.filter);
} else if (!_unselectOnReclick)
{
_selectedFilterBtns[index].selected = true;
} else
{
_selectedFilterBtns.splice(index, 1);
_filter.splice(index, 1);
}

if (index == -1 || _unselectOnReclick)
onSelectionChanged.dispatch(_filter);
}

public function clearSelections():void
{
if (_selectedFilterBtns.length > 0)
{
_selectedFilterBtns[0].selected = false;
_selectedFilterBtns.length = 0;
_filter.length = 0;
}
}

protected function onFilterClicked( e:MouseEvent ):void
{
var selected:FilterButton;
if (e.target is FilterButton)
selected = FilterButton(e.target);
else
selected = FilterButton(e.target.parent);

selectFilterBtn(selected);
}

public function get enabled():Boolean { return enabled; }
public function set enabled( value:Boolean ):void { enabled = value; }

public function clearFilters():void
{
_selectedFilterBtns.length = 0;
_filter.length = 0;
var len:uint = _filterBtns.length;
for
```

```
(var i:uint = 0; i < len; ++i)
{
_filterBtns[i].removeEventListener(MouseEvent.CLICK, onFilterClicked);
_filterBtns[i].destroy();
_filterBtns[i] = null;
}
_filterBtns.length = 0;
}

public function destroy():void
{
clearFilters();
}

}
}
```

File 691: igw\com\ui\core\component\filterlist\FilterListWithHeaders.as

```
package com.ui.core.component.filterlist
{
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.events.MouseEvent;

import org.osflash.signals.Signal;

public class FilterListWithHeaders extends Sprite implements IComponent
{
private var _headers:Vector.<FilterHeader>;

private var _padding:Number;
private var _headerLessBtns:Vector.<FilterButton>;
private var _filterBtns:Vector.<FilterButton>
private var _selectedFilterBtns:Vector.<FilterButton>;
private var _filter:Array;

private var _onlyOneFilterSelected:Boolean;
private var _unselectOnReclick:Boolean;
private var _isHorizontal:Boolean;
private var _enabled:Boolean;
public var onSelectionChanged:Signal;


public function FilterListWithHeaders()
{
_headers = new Vector.<FilterHeader>;
_headerLessBtns
```

```
= new Vector.<FilterButton>;
_filterBtns = new Vector.<FilterButton>;
_selectedFilterBtns = new Vector.<FilterButton>;
_filter = new Array;
onSelectionChanged = new Signal(Array);
super();
}

public function init( bg:String = '', onlyOneFilterSelected:Boolean = true,
unselectOnReclick:Boolean = true):void
{
_onlyOneFilterSelected = onlyOneFilterSelected;
_unselectOnReclick = unselectOnReclick;
}

public function addFilterBtn( filter:*, btn:BitmapButton, padding:Number = 0, index:int = -1,
headerIndex:int = -1 ):void
{
if(index == -1)
index = _filterBtns.length;

var newFilterBtn:FilterButton = new FilterButton(filter, index, btn, padding);
newFilterBtn.addEventListener(MouseEvent.CLICK, onFilterClicked, false, 0, true);
_filterBtns.push(newFilterBtn);
addChild(newFilterBtn);

if(headerIndex == -1)
{
if(_headers.length < 1)
_headerLessBtns.push(newFilterBtn);
else
headerIndex = _headers.length - 1;
}


if(headerIndex != -1 && _headers.length > headerIndex)
{
_headers[headerIndex].addFilter(newFilterBtn);
if(_headers[headerIndex].filterCount > 1)
{
_headers[headerIndex].sortFilters(orderItems);
}
}

layout();
}

public function addFilterHeader( text:Label, bg:String, padding:Number = 0,
headerPadding:Number = 0, index:int = -1 ):void
{
if(index
```

```
== -1)
index = _headers.length;

var newFilterHeader:FilterHeader = new FilterHeader(text, bg, index, padding, headerPadding);
addChild(newFilterHeader);
_headers.push(newFilterHeader);

_headers.sort(orderItems);

layout();
}

private function orderItems( itemOne:*, itemTwo:* ):int
{
var sortOrderOne:Number = itemOne.index;
var sortOrderTwo:Number = itemTwo.index;

if (sortOrderOne < sortOrderTwo)
{
return -1;
} else if (sortOrderOne > sortOrderTwo)
{
return 1;
} else
{
return 0;
}
}

private function layout():void
{
var len:uint;
var filterBtnLen:uint;
var currentFilterHeader:FilterHeader;
var currentFilterBtn:FilterButton;
var xPos:Number = 0;
var yPos:Number = 0;
var i:uint;

len = _headerLessBtns.length;
for (i = 0; i < len; ++i)
{
currentFilterBtn = _headerLessBtns[i];
currentFilterBtn.x = xPos;
currentFilterBtn.y = yPos;
yPos += currentFilterBtn.height + currentFilterBtn.padding;
}

len = _headers.length;
for(i = 0; i < len; ++i)
{
```

```
currentFilterHeader = _headers[i];
currentFilterHeader.x = xPos;
currentFilterHeader.y = yPos;
filterBtnLen = currentFilterHeader.filterBtns.length;
yPos += currentFilterHeader.height + currentFilterHeader.padding;
for( var j:uint = 0; j < filterBtnLen; ++j)
{
currentFilterBtn = currentFilterHeader.filterBtns[j];
currentFilterBtn.x = xPos;
currentFilterBtn.y = yPos;
yPos += currentFilterBtn.height + currentFilterBtn.padding;
}

yPos += currentFilterHeader.headerPadding;
}
}

public function selectFilterByIndex( index:int ):void
{
if(index < _filterBtns.length)
{
selectFilterBtn(_filterBtns[index]);
_filterBtns[index].selected = true;
}
}

private function selectFilterBtn( filterBtn:FilterButton ):void
{
var index:int = _selectedFilterBtns.indexOf(filterBtn);

if(_onlyOneFilterSelected && _selectedFilterBtns.length > 0)
{
if(index == -1 && _unselectOnReclick == false)
clearSelections();
}

if(index == -1)
{
_selectedFilterBtns.push(filterBtn);
_filter.push(filterBtn.filter);
}
else if(!_unselectOnReclick)
{
_selectedFilterBtns[index].selected = true;
}
else
{
_selectedFilterBtns.splice(index, 1);
_filter.splice(index, 1);
}
```

```
if(index == -1 || _unselectOnReclick)
onSelectionChanged.dispatch(_filter);
}

public function clearSelections():void
{
if(_selectedFilterBtns.length > 0)
{
_selectedFilterBtns[0].selected = false;
_selectedFilterBtns.length = 0;
_filter.length = 0;
}
}

protected function onFilterClicked( e:MouseEvent ):void
{
var selected:FilterButton;
if( e.target is FilterButton )
selected = FilterButton(e.target);
else
selected = FilterButton(e.target.parent);

selectFilterBtn(selected);
}

public function get enabled():Boolean { return enabled; }
public function set enabled( value:Boolean ):void
{
_enabled = value;
var len:uint = _filterBtns.length;
for(var i:uint = 0; i < len; ++i)
{
_filterBtns[i].enabled = _enabled;
}
}

public function destroy():void
{
_selectedFilterBtns.length = 0;
_filter.length = 0;
var len:uint = _filterBtns.length;
for(var i:uint = 0; i < len; ++i)
{
_filterBtns[i].removeEventListener(MouseEvent.CLICK, onFilterClicked);
_filterBtns[i].destroy();
_filterBtns[i] = null;
}
_filterBtns.length = 0;
}
```

```
}
}
-----------------------------------
File 692: igw\com\ui\core\component\label\Label.as
package com.ui.core.component.label
{
import com.service.language.Localization;

import flash.events.FocusEvent;
import flash.text.AntiAliasType;
import flash.text.GridFitType;
import flash.text.TextField;
import flash.text.TextFieldAutoSize;
import flash.text.TextFieldType;
import flash.text.TextFormat;
import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;
import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;

import com.service.ExternalInterfaceAPI;

import com.StartupConfig;

public class Label extends TextField
{
public static var showUnlocalizedString:Boolean;

protected var _constrict:Boolean;
protected var _desiredFontSize:Number;
protected var _desiredWidth:Number;
protected var _desiredHeight:Number;
protected var _inputMessage:String;
protected var _textFormat:TextFormat;
protected var _clearOnFocusIn:Boolean;
protected var _useLocalization:Boolean;
protected var _allCaps:Boolean;

private var _unlocalizedString:String;
private var _tokens:Object;
private var _labelColor:LabelColor;

private const logger:ILogger = getLogger('ui.core.component.Label');

public function Label( fontSize:Number = 12, color:uint = 0, maxWidth:Number = 100,
maxHeight:Number = 20, useLocalization:Boolean = true, fontNr:int = 0 )
{
//ExternalInterfaceAPI.logConsole("Use
```

```
font: " + font);
init(fontSize, color, maxWidth, maxHeight, useLocalization, fontNr);
}

public function init( fontSize:Number = 12, color:uint = 0, maxWidth:Number = 100,
maxHeight:Number = 20, useLocalization:Boolean = true, fontNr:int = 0 ):void
{
embedFonts = true;
selectable = false;
mouseEnabled = multiline = false;
_constrict = true;
textColor = color;
antiAliasType = AntiAliasType.ADVANCED;
gridFitType = GridFitType.SUBPIXEL;
_desiredFontSize = fontSize;
_desiredWidth = maxWidth;
_desiredHeight = maxHeight;
width = maxWidth;
height = maxHeight;
_textFormat = new TextFormat(StartupConfig.FontArray[fontNr]);
defaultTextFormat = _textFormat;
this.fontSizeFormat = fontSize;
align = TextFormatAlign.CENTER;
_useLocalization = useLocalization;
text = '';
}

private function localizeText( value:String ):String
{
var localizedString:String = _unlocalizedString = value;
if (!showUnlocalizedString && value != '' && _useLocalization)
{
localizedString = Localization.instance.getString(_unlocalizedString);
if (localizedString == '' && _unlocalizedString != '')
{
localizedString = _unlocalizedString;
if (isNaN(Number(value)))
logger.debug('Unlocalized String = {}', _unlocalizedString);
}
}

if (_allCaps)
localizedString = localizedString.toUpperCase();

return localizedString;
}

private function localizeTextWithTokens( value:String, tokens:Object ):String
{
var localizedString:String = _unlocalizedString = value;
_tokens
```

```
= tokens;
if (!showUnlocalizedString)
{
localizedString = Localization.instance.getStringWithTokens(_unlocalizedString, tokens);

if (localizedString == '' && _unlocalizedString != '')
{
localizedString = value;
logger.debug('Unlocalized String = {}', _unlocalizedString);
}
}
return localizedString;
}

public function setTextWithTokens( value:String, tokens:Object ):void
{
super.text = localizeTextWithTokens(value, tokens);
if (_constrict)
resize();
}

public function setHtmlTextWithTokens( value:String, tokens:Object ):void
{
super.htmlText = localizeTextWithTokens(value, tokens);
if (_constrict)
resize();
}

public function setBuildTime( seconds:Number, maxElems:int = 4 ):void
{
super.text = StringUtil.getBuildTime(seconds, maxElems);
if (_constrict)
resize();
}

public function setSize( w:Number, h:Number ):void
{
_desiredWidth = w;
_desiredHeight = h;
resize();
}

protected function resize():void
{
fontSizeFormat = _desiredFontSize;
autoSize = TextFieldAutoSize.LEFT;
var value:String = text;
var firstRun:Boolean = true;
do
{
if
```

```
(!firstRun)
fontSizeFormat--;
if (fontSizeFormat == 0)
break;
super.text = value;
if (multiline)
width = _desiredWidth;
firstRun = false;
} while (width > _desiredWidth || height > _desiredHeight)
autoSize = TextFieldAutoSize.NONE;
width = _desiredWidth;
height = _desiredHeight;
}

public function addLabelColor( selectionColor:uint = 0x000000, selectedColor:uint = 0x000000
):void
{
_labelColor = new LabelColor(this, textColor, selectionColor, selectedColor);
}

private function updateFocusListeners():void
{
if (type == TextFieldType.INPUT)
{
if (_clearOnFocusIn)
{
removeEventListener(FocusEvent.FOCUS_IN, onFocusIn);
removeEventListener(FocusEvent.FOCUS_OUT, onFocusOut);
addEventListener(FocusEvent.FOCUS_IN, onFocusIn, false, 0, true);
addEventListener(FocusEvent.FOCUS_OUT, onFocusOut, false, 0, true)
} else
{
removeEventListener(FocusEvent.FOCUS_IN, onFocusIn);
removeEventListener(FocusEvent.FOCUS_OUT, onFocusOut);
}
}
}

public function updateLabelColor( value:uint ):void
{
if (_labelColor)
_labelColor.textColor = value;
}

public function updateLabelSelectedColor( value:uint ):void
{
if (_labelColor)
_labelColor.selectedColor = value;
}

public
```

```
function updateLabelSelectionColor( value:uint ):void
{
if (_labelColor)
_labelColor.selectionColor = value;
}

private function onFocusIn( e:FocusEvent ):void
{
if (text == _inputMessage)
text = '';
}

private function onFocusOut( e:FocusEvent ):void
{
if (text == "")
{
super.text = _inputMessage;
if (_constrict)
resize();
}
}

public function get align():String { return defaultTextFormat.align; }
public function set align( type:String ):void
{
_textFormat.align = type;
defaultTextFormat = _textFormat;
}

public function get allCaps():Boolean { return _allCaps; }
public function set allCaps( value:Boolean ):void { _allCaps = value; }

public function set allowInput( v:Boolean ):void
{
type = (v) ? TextFieldType.INPUT : TextFieldType.DYNAMIC;
selectable = v;
mouseEnabled = v;
constrictTextToSize = !v;
align = TextFormatAlign.LEFT;
width = _desiredWidth;
height = _desiredHeight;
_inputMessage = text;
text = _inputMessage;
updateFocusListeners();
}

public function updateInputText( v:String ):void
{
if (text == _inputMessage)
text = "";

if
```

```
(useLocalization)
_inputMessage = localizeText(v);
else
_inputMessage = v;

if (text == "")
{
super.text = _inputMessage;
if (_constrict)
resize();
}
}

public function get inputMessage():String { return _inputMessage; }

public function set bold( value:Boolean ):void { _textFormat.bold = value; defaultTextFormat =
_textFormat; }

public function set clearOnFocusIn( v:Boolean ):void { _clearOnFocusIn = v;
updateFocusListeners(); }

public function set constrictTextToSize( v:Boolean ):void { _constrict = v; }

public function set fontSize( v:int ):void { _desiredFontSize = v; resize(); }

protected function get fontSizeFormat():* { return _textFormat.size; }
protected function set fontSizeFormat( value:int ):void
{
_textFormat.size = value;
defaultTextFormat = _textFormat;
}

override public function set htmlText( value:String ):void
{
super.htmlText = localizeText(value);
if (_constrict)
resize();
}

public function set letterSpacing( value:Number ):void
{
_textFormat.letterSpacing = value;
defaultTextFormat = _textFormat;
text = text;
}
public function set leading( value:int ):void { _textFormat.leading = value; defaultTextFormat =
_textFormat; }
public function set leftMargin( margin:int ):void { _textFormat.leftMargin = margin;
defaultTextFormat = _textFormat; }

override
```

```
public function set multiline( value:Boolean ):void { wordWrap = value; super.multiline = value; }

public function set rightMargin( margin:int ):void { _textFormat.rightMargin = margin;
defaultTextFormat = _textFormat; }

override public function set text( value:String ):void
{
if(!value)
value = "";
super.text = localizeText(value);
if (_constrict)
resize();
}

public function set underline( value:Boolean ):void { _textFormat.underline = value;
defaultTextFormat = _textFormat; }

public function get useLocalization():Boolean { return _useLocalization; }
public function set useLocalization( useLoc:Boolean ):void { _useLocalization = useLoc; }

public function get unLocalizedText():String { return _unlocalizedString; }

public function destroy():void
{
type = TextFieldType.DYNAMIC;
autoSize = TextFieldAutoSize.NONE;
filters = [];
_clearOnFocusIn = false;
_labelColor = null;
_textFormat = null;
_tokens = null;
styleSheet = null;
updateFocusListeners();
super.htmlText = super.text = '';
}
}
}

------------------------------------
File 693: igw\com\ui\core\component\label\LabelColor.as
package com.ui.core.component.label
{
import com.ui.core.component.label.Label;
import flash.filters.ColorMatrixFilter;

/**
* @author Keita
* labs.hellokeita.com
*/

public
```

```
class LabelColor
{

private static const byteToPerc:Number = 1 / 0xff;

private var _textField:Label;
private var _textColor:uint;
private var _selectedColor:uint;
private var _selectionColor:uint;
private var colorMatrixFilter:ColorMatrixFilter;

public function LabelColor( textField:Label, textColor:uint = 0x000000, selectionColor:uint =
0x000000, selectedColor: uint = 0x000000 )
{
_textField = textField;

colorMatrixFilter = new ColorMatrixFilter();
_textColor = textColor;
_selectionColor = selectionColor;
_selectedColor = selectedColor;
updateFilter();
}

public function set textField( tf:Label ):void
{
_textField = tf;
}
public function get textField():Label
{
return _textField;
}
public function set textColor( c:uint ):void
{
_textColor = c;
updateFilter();
}
public function get textColor():uint
{
return _textColor;
}
public function set selectionColor( c:uint ):void
{
_selectionColor = c;
updateFilter();
}
public function get selectionColor():uint
{
return _selectionColor;
}
public function set selectedColor( c:uint ):void
{
```

```
_selectedColor = c;
updateFilter();
}
public function get selectedColor():uint
{
return _selectedColor;
}

private function updateFilter():void
{

_textField.textColor = 0xff0000;

var o:Array = splitRGB(_selectionColor);
var r:Array = splitRGB(_textColor);
var g:Array = splitRGB(_selectedColor);

var ro:int = o[0];
var go:int = o[1];
var bo:int = o[2];

var rr:Number = ((r[0] - 0xff) - o[0]) * byteToPerc + 1;
var rg:Number = ((r[1] - 0xff) - o[1]) * byteToPerc + 1;
var rb:Number = ((r[2] - 0xff) - o[2]) * byteToPerc + 1;

var gr:Number = ((g[0] - 0xff) - o[0]) * byteToPerc + 1 - rr;
var gg:Number = ((g[1] - 0xff) - o[1]) * byteToPerc + 1 - rg;
var gb:Number = ((g[2] - 0xff) - o[2]) * byteToPerc + 1 - rb;

colorMatrixFilter.matrix = [rr, gr, 0, 0, ro, rg, gg, 0, 0, go, rb, gb, 0, 0, bo, 0, 0, 0, 1, 0];

_textField.filters = [colorMatrixFilter];

}

private static function splitRGB( color:uint ):Array
{
return [color >> 16 & 0xff, color >> 8 & 0xff, color & 0xff];
}
}
}
```

File 694: igw\com\ui\core\component\label\LabelFactory.as

```
package com.ui.core.component.label
{
import flash.text.StyleSheet;
import com.StartupConfig;

public class LabelFactory
{
```

```
public static const TITLE_COLOR:int = 0xffffff //0xb4e0ff;
public static const DYNAMIC_TEXT_COLOR:int = 0xf0f0f0;
public static const LINK_TEXT_COLOR:int = 0xf04c4c;
public static const TOAST_GOLD:int = 0xfac973;
public static const TRAY_TITLE_COLOR:int = 0xc6e2f2;
public static const SPEED_UP_COLOR:int = 0xf7c78b;

//public static const TITLE_FONT:String = "Agency FB";
//public static const BODY_FONT:String = "Open Sans";

public static const LABEL_TYPE_TITLE:int = 0;
public static const LABEL_TYPE_HEADER:int = 1;
public static const LABEL_TYPE_DYNAMIC:int = 2;
public static const LABEL_TYPE_DIALOG_TITLE:int = 3;
public static const LABEL_TYPE_TRAY_TITLE:int = 4;
public static const LABEL_TYPE_SPEED_UP:int = 5;

private static var _linkStyleSheet:StyleSheet;

public static function createLabel( labelType:int, width:Number, size:int = -1, col:int = -1,
multiline:Boolean = false ):Label
{
var allCaps:Boolean;
var color:int = col == -1 ? TITLE_COLOR : col;
var font:int = StartupConfig.FontArray[0];

switch (labelType)
{
case LABEL_TYPE_TITLE:
size = size == -1 ? 18 : size;
break;

case LABEL_TYPE_HEADER:
size = size == -1 ? 14 : size;
allCaps = true;
break;

case LABEL_TYPE_DYNAMIC:
size = size == -1 ? 12 : size;
color = col == -1 ? DYNAMIC_TEXT_COLOR : col;
font = StartupConfig.FontArray[1];
break;

case LABEL_TYPE_DIALOG_TITLE:
size = size == -1 ? 24 : size;
color = DYNAMIC_TEXT_COLOR;
allCaps = true;
break;

case
```

```
LABEL_TYPE_TRAY_TITLE:
size = size == -1 ? 18 : size;
color = TRAY_TITLE_COLOR;
allCaps = true;
break;

case LABEL_TYPE_SPEED_UP:
size = size == -1 ? 11 : size;
color = SPEED_UP_COLOR;
font = StartupConfig.FontArray[1];
allCaps = true;
break;

default:

break;
}

var result:Label = new Label(size, color, width, size * 1.6, true, font);
result.allCaps = allCaps;
result.multiline = multiline;
return result;
}

public static function get linkStyleSheet():StyleSheet
{
if (!_linkStyleSheet)
{
var link:Object = {color:"#" + LINK_TEXT_COLOR.toString(16), textDecoration:"underline"};
_linkStyleSheet = new StyleSheet();
_linkStyleSheet.setStyle("a:link", link);
_linkStyleSheet.setStyle("a:hover", {color:"#" + LINK_TEXT_COLOR.toString(16),
textDecoration:'underline'});
}

return _linkStyleSheet;
}
}
}

-------------------------------------
File 695: igw\com\ui\core\component\label\RequirementLabel.as
package com.ui.core.component.label
{
import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.utils.getDefinitionByName;

import com.StartupConfig;

public
```

```
class RequirementLabel extends Sprite
{
private var _bullet:Bitmap;
private var _checkMark:Bitmap;

private var _lbl:Label;

private var _showLink:Boolean;

public function RequirementLabel( dx:int = 0, dy:int = 0, fontSize:Number = 12, color:uint = 0,
maxWidth:Number = 100, maxHeight:Number = 20, useLocalization:Boolean = true, fontNr:int =
0 )
{
_lbl = new Label(fontSize, color, maxWidth, maxHeight, useLocalization,
StartupConfig.FontArray[fontNr]);
_lbl.x = 14; //dx;
_lbl.y = dy;
_lbl.useLocalization = false;
addChild(_lbl);

var checkMarkClass:Class = Class(getDefinitionByName('CheckMarkBMD'));
var bulletClass:Class = Class(getDefinitionByName('BulletBMD'));

_checkMark = new Bitmap(BitmapData(new checkMarkClass()))
// _checkMark.x = _lbl.x - 5;
// _checkMark.y = _lbl.y;
_checkMark.visible = true;
addChild(_checkMark);

_bullet = new Bitmap(BitmapData(new bulletClass()))
// _bullet.x = _lbl.x - 14;
// _bullet.y = _lbl.y;
_bullet.visible = true;
addChild(_bullet);

_showLink = false;
}

public function formatObject():void
{
_bullet.x = _lbl.x - 14;
_bullet.y = _lbl.y + _lbl.textHeight * 0.5 - 2;

_checkMark.x = _lbl.x + _lbl.textWidth + 15;
_checkMark.y = _lbl.y + _lbl.textHeight - _checkMark.height;

}

public function get lbl():Label { return _lbl; }
public function set lbl( value:Label ):void { _lbl = value; }

public
```

```
function get checkMark():Bitmap { return _checkMark; }

public function get showLink():Boolean { return _showLink; }
public function set showLink( value:Boolean ):void { _showLink = value; }

public function destroy():void
{
_bullet = null;
_checkMark = null;

_lbl.destroy();
_lbl = null;
}

}
}

-----------------------------------
File 696: igw\com\ui\core\component\misc\ActionComponent.as
package com.ui.core.component.misc
{
import com.controller.transaction.requirements.RequirementVO;
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.ButtonPrototype;
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

public class ActionComponent extends Sprite implements IComponent
{
private var _actionBtn:BitmapButton;
private var _instantActionBtn:BitmapButton;

private var _cannotAffordActionBtn:BitmapButton;
private var _cannotAffordInstantActionBtn:BitmapButton;

private var _cannotAffordActionBtnProto:ButtonPrototype;
private var _cannotAffordInstantActionBtnProto:ButtonPrototype;

private var _actionBtnProto:ButtonPrototype;
private var _instantActionBtnProto:ButtonPrototype;

private
```

```
var _actionTimeBox:Bitmap;
private var _instantActionCostBox:Bitmap;
private var _cannotAffordActionTimeBox:Bitmap;

private var _instantCost:Label;
private var _timeCost:Label;

private var _premiumSymbol:Bitmap;

private var _getMoreResourcesPremiumSymbol:Bitmap;

private var _enabled:Boolean;

private var _requirements:RequirementVO;

private var _enabledFontSize:int;
private var _disabledFontSize:int;
private var _enabledLabelYPos:int;

private var _freeText:String = 'CodeString.Shared.Free'; //Free
private var _offlineString:String = 'CodeString.Shipyard.BtnStatus.Offline'; //OFFLINE

public function ActionComponent( actionButton:ButtonPrototype,
instantActionButton:ButtonPrototype, cannotAffordActionBtnProto:ButtonPrototype,
cannotAffordInstantActionButton:ButtonPrototype,
instantActionCost:int = 0, ActionTimeCost:int = 0 )
{
super();

_enabledFontSize = 20;
_disabledFontSize = 24;

var premiumSymbolClass:Class = Class(getDefinitionByName(('KalganSymbolBMD')));

_actionBtnProto = actionButton;
_instantActionBtnProto = instantActionButton;

_cannotAffordActionBtnProto = cannotAffordActionBtnProto;
_cannotAffordInstantActionBtnProto = cannotAffordInstantActionButton;

_instantActionBtn = UIFactory.getButton(ButtonEnum.GOLD_A, 126, 55, 134, 0,
instantActionButton.text);
_instantActionBtn.addEventListener(MouseEvent.CLICK, instantActionButton.callback, false, 0,
true);
_instantActionBtn.fontSize = _enabledFontSize;
_instantActionBtn.label.y -= 4;

_actionBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 126, 55, 0, 0, actionButton.text);
_actionBtn.addEventListener(MouseEvent.CLICK, actionButton.callback, false, 0, true);
_actionBtn.fontSize = _enabledFontSize;
_actionBtn.label.y
```

```
-= 4;

_cannotAffordInstantActionBtn = UIFactory.getButton(ButtonEnum.GOLD_A, 126, 55, 134, 0,
cannotAffordInstantActionButton.text);
_cannotAffordInstantActionBtn.addEventListener(MouseEvent.CLICK,
cannotAffordInstantActionButton.callback, false, 0, true);
_cannotAffordInstantActionBtn.fontSize = _enabledFontSize;
_cannotAffordInstantActionBtn.label.y -= 4;
_cannotAffordInstantActionBtn.visible = false;

_cannotAffordActionBtn = UIFactory.getButton(ButtonEnum.GOLD_A, 126, 55, 0, 0,
cannotAffordActionBtnProto.text)
_cannotAffordActionBtn.addEventListener(MouseEvent.CLICK,
cannotAffordActionBtnProto.callback, false, 0, true);
_cannotAffordActionBtn.fontSize = _enabledFontSize;
_cannotAffordActionBtn.label.y -= 4;
_cannotAffordActionBtn.visible = false;

_enabledLabelYPos = _actionBtn.label.y;

_premiumSymbol = new Bitmap(BitmapData(new premiumSymbolClass()));
_premiumSymbol.x = 140;
_premiumSymbol.y = 28;

_timeCost = new Label(18, 0xf0f0f0, 119, 21, false);
_timeCost.align = TextFormatAlign.CENTER;
_timeCost.constrictTextToSize = false;
_timeCost.x = 4;
_timeCost.y = 28;
_timeCost.text = String(instantActionCost);

_instantCost = new Label(18, 0xf0f0f0, 119, 25, false);
_instantCost.align = TextFormatAlign.CENTER;
_instantCost.constrictTextToSize = false;
_instantCost.x = 135;
_instantCost.y = 28;
_instantCost.setBuildTime(ActionTimeCost);

_actionTimeBox = UIFactory.getScaleBitmap('InputBoxBMD');
_actionTimeBox.width = 119;
_actionTimeBox.height = 22;
_actionTimeBox.x = 4;
_actionTimeBox.y = 28;

_instantActionCostBox = UIFactory.getScaleBitmap('InputBoxGoldBMD');
_instantActionCostBox.width = 119;
_instantActionCostBox.height = 22;
_instantActionCostBox.x = 135;
_instantActionCostBox.y = 28;

_cannotAffordActionTimeBox
```

```
= UIFactory.getScaleBitmap('InputBoxGoldBMD');
_cannotAffordActionTimeBox.width = 119;
_cannotAffordActionTimeBox.height = 22;
_cannotAffordActionTimeBox.x = 4;
_cannotAffordActionTimeBox.y = 28;
_cannotAffordActionTimeBox.visible = false;

addChild(_instantActionBtn);
addChild(_actionBtn);

addChild(_cannotAffordInstantActionBtn);
addChild(_cannotAffordActionBtn);

addChild(_actionTimeBox);
addChild(_instantActionCostBox);
addChild(_cannotAffordActionTimeBox);

addChild(_timeCost);
addChild(_instantCost);

addChild(_premiumSymbol);

_enabled = true;
}

public function set actionBtnText( text:String ):void
{
_actionBtn.text = text;

if (!enabled)
_actionBtn.label.y += 6;
}

public function set instantActionBtnText( text:String ):void
{
_instantActionBtn.text = text;

if (!enabled)
_instantActionBtn.label.y += 6;
}

public function set instantCost( instantActionCost:int ):void
{
if (instantActionCost > 0)
{
_instantCost.useLocalization = false;
_instantCost.text = String(instantActionCost);
} else
{
_instantCost.useLocalization = true;
_instantCost.text
```

```
= _freeText;
}

}

public function set timeCost( actionTimeCost:int ):void
{
_timeCost.setBuildTime(actionTimeCost);
}

public function set requirements( reqs:RequirementVO ):void
{
_requirements = reqs;

if (_enabled)
updateBasedOnRequirements();
}

private function updateBasedOnRequirements():void
{
if (_requirements)
{
if (!_requirements.purchaseVO.costExceedsMaxResources)
{
_instantActionBtn.visible = _requirements.purchaseVO.canPurchaseWithPremium;
_cannotAffordInstantActionBtn.visible = !_requirements.purchaseVO.canPurchaseWithPremium;

_actionBtn.visible = _requirements.purchaseVO.canPurchase;
_cannotAffordActionBtn.visible = !_requirements.purchaseVO.canPurchase;
_cannotAffordActionTimeBox.visible = !_requirements.purchaseVO.canPurchase;
}

}
}

public function set enabled( value:Boolean ):void
{
_enabled = value;

instantActionBtnEnabled = _enabled;
actionBtnEnabled = _enabled;

cannotAffordInstantActionBtnEnabled = _enabled;
cannotAffordActionBtnEnabled = _enabled;

if (_enabled)
updateBasedOnRequirements();
}

public function get enabled():Boolean { return _enabled; }

public
```

```
function set instantActionBtnEnabled( enabled:Boolean ):void
{
_instantActionBtn.enabled = enabled;
_premiumSymbol.visible = enabled;
_instantCost.visible = enabled;
_instantActionCostBox.visible = enabled;
if (!enabled)
{
_instantActionBtn.fontSize = _disabledFontSize;
_instantActionBtn.text = _offlineString;
_instantActionBtn.label.y += 6;
} else
{
_instantActionBtn.fontSize = _enabledFontSize;
_instantActionBtn.text = _instantActionBtnProto.text;
_instantActionBtn.label.y = _enabledLabelYPos;
}
}

public function set actionBtnEnabled( enabled:Boolean ):void
{
_actionBtn.enabled = enabled;
_timeCost.visible = enabled;
_actionTimeBox.visible = enabled;
if (!enabled)
{
_actionBtn.fontSize = _disabledFontSize;
_actionBtn.text = _offlineString;
_actionBtn.label.y += 6;
} else
{
_actionBtn.fontSize = _enabledFontSize;
_actionBtn.text = _actionBtnProto.text;
_actionBtn.label.y = _enabledLabelYPos;

}
}

public function set cannotAffordInstantActionBtnEnabled( enabled:Boolean ):void
{
_cannotAffordInstantActionBtn.enabled = enabled;
_cannotAffordActionTimeBox.visible = enabled;
if (!enabled)
{
_cannotAffordInstantActionBtn.fontSize = _disabledFontSize;
_cannotAffordInstantActionBtn.text = _offlineString;
_cannotAffordInstantActionBtn.label.y += 6;
} else
{
_cannotAffordInstantActionBtn.fontSize = _enabledFontSize;
_cannotAffordInstantActionBtn.text
```

```
= _cannotAffordInstantActionBtnProto.text;
_cannotAffordInstantActionBtn.label.y = _enabledLabelYPos;
}
}

public function set cannotAffordActionBtnEnabled( enabled:Boolean ):void
{
_cannotAffordActionBtn.enabled = enabled;
_instantActionCostBox.visible = enabled;
if (!enabled)
{
_cannotAffordActionBtn.fontSize = _disabledFontSize;
_cannotAffordActionBtn.text = _offlineString;
_cannotAffordActionBtn.label.y += 6;
} else
{
_cannotAffordActionBtn.fontSize = _enabledFontSize;
_cannotAffordActionBtn.text = _cannotAffordActionBtnProto.text;
_cannotAffordActionBtn.label.y = _enabledLabelYPos;
}
}

public function get actionBtn():BitmapButton { return _actionBtn; }
public function get instantActionBtn():BitmapButton { return _instantActionBtn; }

public function destroy():void
{
_instantActionBtn.removeEventListener(MouseEvent.CLICK, _instantActionBtnProto.callback);
_instantActionBtn.destroy();
_instantActionBtn = null;
_instantActionBtnProto = null;

_actionBtn.removeEventListener(MouseEvent.CLICK, _actionBtnProto.callback);
_actionBtn.destroy();
_actionBtn = null;
_actionBtnProto = null;

_timeCost.destroy();
_timeCost = null;

_instantCost.destroy();
_instantCost = null;

_premiumSymbol = null;
}
}
}

-----------------------------------
File 697: igw\com\ui\core\component\misc\ActionInProgressComponent.as
package
```

```
com.ui.core.component.misc
{
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.ButtonPrototype;
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

public class ActionInProgressComponent extends Sprite implements IComponent
{
private var _cancelBtn:BitmapButton;
private var _speedUpBtn:BitmapButton;

private var _cancelBtnProto:ButtonPrototype;
private var _speedUpBtnProto:ButtonPrototype;

private var _timeRemainingBG:Bitmap;

private var _timeRemaining:Label;

private var _enabled:Boolean;

private var _pendingText:String = 'CodeString.Shared.Pending';

public function ActionInProgressComponent( cancelButton:ButtonPrototype,
speedUpButton:ButtonPrototype, timeRemaining:int = 0 )
{
super();

var timeRemainingBGClass:Class = Class(getDefinitionByName('BuildRepairTimerBGBMD'));

_cancelBtnProto = cancelButton;
_speedUpBtnProto = speedUpButton;

_speedUpBtn = UIFactory.getButton(ButtonEnum.GOLD_A, 126, 55, 0, 0,
_speedUpBtnProto.text);
_speedUpBtn.label.y += 4;
_speedUpBtn.addEventListener(MouseEvent.CLICK, _speedUpBtnProto.callback, false, 0,
true);

_cancelBtn
```

```
= UIFactory.getButton(ButtonEnum.RED_A, 134, 29, _speedUpBtn.width + 5, 27,
_cancelBtnProto.text);

if (_cancelBtnProto.callback != null)
_cancelBtn.addEventListener(MouseEvent.CLICK, _cancelBtnProto.callback, false, 0, true);

_timeRemainingBG = UIFactory.getScaleBitmap('InputBoxBMD');
_timeRemainingBG.width = 134;
_timeRemainingBG.height = 23;
_timeRemainingBG.y = 1;
_timeRemainingBG.x = _speedUpBtn.width + 5;

_timeRemaining = new Label(16, 0xffffff, _timeRemainingBG.width, _timeRemainingBG.height);
_timeRemaining.align = TextFormatAlign.CENTER;
_timeRemaining.x = _timeRemainingBG.x;
_timeRemaining.y = _timeRemainingBG.y;
_timeRemaining.setBuildTime(timeRemaining);

addChild(_cancelBtn);
addChild(_speedUpBtn);

addChild(_timeRemainingBG);
addChild(_timeRemaining);
}

public function addCancelBtnCallback( callback:Function ):void
{
_cancelBtn.addEventListener(MouseEvent.CLICK, callback, false, 0, true);
}

public function set timeRemaining( time:int ):void
{
_cancelBtn.enabled = time > 0;
_speedUpBtn.enabled = time > 0;
if (time > 0)
{
_cancelBtn.filters = []
_speedUpBtn.filters = [];
} else
{
_cancelBtn.filters = [CommonFunctionUtil.getGreyScaleFilter()];
_speedUpBtn.filters = [CommonFunctionUtil.getGreyScaleFilter()];
}
if (time < 0)
_timeRemaining.text = _pendingText;
else
_timeRemaining.setBuildTime(time / 1000);
}

public function set enabled( value:Boolean ):void
{
```

```
_enabled = value;
_cancelBtn.enabled = _enabled;
_speedUpBtn.enabled = _enabled;
}

public function get enabled():Boolean { return _enabled; }

public function destroy():void
{
if (_cancelBtnProto.callback != null)
_cancelBtn.removeEventListener(MouseEvent.CLICK, _cancelBtnProto.callback);

_cancelBtn.destroy();
_cancelBtn = null;
_cancelBtnProto = null;

if (_speedUpBtnProto.callback != null)
_speedUpBtn.removeEventListener(MouseEvent.CLICK, _speedUpBtnProto.callback);

_speedUpBtn.destroy();
_speedUpBtn = null;
_speedUpBtnProto = null;

_timeRemaining.destroy();
_timeRemaining = null;

_timeRemainingBG = null;
}

}
}

-----------------------------------
File 698: igw\com\ui\core\component\misc\BlueprintActionComponent.as
package com.ui.core.component.misc
{
import com.model.player.CurrentUser;
import com.ui.core.ButtonPrototype;
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

public
```

```
class BlueprintActionComponent extends Sprite implements IComponent
{
private var _partialPurchaseBtn:BitmapButton;
private var _fullPurchaseBtn:BitmapButton;

private var _cannotFullPurchaseBtn:BitmapButton;
private var _cannotAffordPartialPurchaseBtn:BitmapButton;

private var _cannotPartialCostBtnProto:ButtonPrototype;
private var _cannotAffordFullCostBtnProto:ButtonPrototype;

private var _partialPurchaseBtnProto:ButtonPrototype;
private var _fullPurchaseBtnProto:ButtonPrototype;

private var _fullCostText:Label;
private var _partialCostText:Label;

private var _leftPremiumSymbol:Bitmap;
private var _rightPremiumSymbol:Bitmap;

private var _enabled:Boolean;

private var _fullCost:int;
private var _partialCost:int;

private var _enabledFontSize:int;
private var _disabledFontSize:int;
private var _enabledLabelYPos:int;

private var _freeText:String = 'CodeString.Shared.Free'; //Free
private var _offlineText:String = 'OFFLINE';

public function BlueprintActionComponent( fullPurchaseButton:ButtonPrototype,
partialPurchaseButton:ButtonPrototype, cannotAffordPartialPurchaseBtnProto:ButtonPrototype,
cannotAffordFullCostButton:ButtonPrototype,
partialCost:int = 0, fullCost:int = 0 )
{
super();

_fullCost = fullCost;
_partialCost = partialCost;

_enabledFontSize = 20;
_disabledFontSize = 24;

_partialPurchaseBtnProto = partialPurchaseButton;
_fullPurchaseBtnProto = fullPurchaseButton;

_cannotPartialCostBtnProto = cannotAffordPartialPurchaseBtnProto;
_cannotAffordFullCostBtnProto = cannotAffordFullCostButton;

var
```

```
premiumBitmapData:BitmapData = PanelFactory.getBitmapData('KalganSymbolBMD');

_partialPurchaseBtn = ButtonFactory.getBitmapButton('InstantBtnNeutralBMD', 0, 0,
partialPurchaseButton.text, 0xd9ab70, 'InstantBtnRollOverBMD', 'InstantBtnDownBMD',
'InactiveInstantBtn');
_partialPurchaseBtn.addEventListener(MouseEvent.CLICK, partialPurchaseButton.callback,
false, 0, true);
_partialPurchaseBtn.fontSize = _enabledFontSize;
_partialPurchaseBtn.label.y -= 3;

_fullPurchaseBtn = ButtonFactory.getBitmapButton('ShortResourcesBtnNeutralBMD', 145, 0,
fullPurchaseButton.text, 0xd9ab70, 'ShortResourcesBtnRolloverBMD',
'ShortResourcesBtnDownBMD', 'InactiveBuildBtn');
_fullPurchaseBtn.addEventListener(MouseEvent.CLICK, fullPurchaseButton.callback, false, 0,
true);
_fullPurchaseBtn.fontSize = _enabledFontSize;
_fullPurchaseBtn.label.y -= 3;

_cannotAffordPartialPurchaseBtn = ButtonFactory.getBitmapButton('InstantBtnNeutralBMD', 0,
0, cannotAffordPartialPurchaseBtnProto.text, 0xd9ab70, 'InstantBtnRollOverBMD',
'InstantBtnDownBMD',
'InactiveInstantBtn');
_cannotAffordPartialPurchaseBtn.addEventListener(MouseEvent.CLICK,
cannotAffordPartialPurchaseBtnProto.callback, false, 0, true);
_cannotAffordPartialPurchaseBtn.fontSize = _enabledFontSize;
_cannotAffordPartialPurchaseBtn.label.y -= 3;
_cannotAffordPartialPurchaseBtn.visible = false;

_cannotFullPurchaseBtn = ButtonFactory.getBitmapButton('ShortResourcesBtnNeutralBMD',
145, 0, cannotAffordFullCostButton.text, 0xd9ab70, 'ShortResourcesBtnRolloverBMD',
'ShortResourcesBtnDownBMD',
'InactiveBuildBtn');
_cannotFullPurchaseBtn.addEventListener(MouseEvent.CLICK,
cannotAffordFullCostButton.callback, false, 0, true);
_cannotFullPurchaseBtn.fontSize = _enabledFontSize;
_cannotFullPurchaseBtn.label.y -= 3;
_cannotFullPurchaseBtn.visible = false;

_enabledLabelYPos = _fullPurchaseBtn.label.y;

_leftPremiumSymbol = new Bitmap(premiumBitmapData);
_leftPremiumSymbol.x = 33;
_leftPremiumSymbol.y = 26;

_rightPremiumSymbol = new Bitmap(premiumBitmapData);
_rightPremiumSymbol.x = _fullPurchaseBtn.x + 36;
_rightPremiumSymbol.y = 26;

_partialCostText = new Label(18, 0xf0f0f0, 50, 25, false);
_partialCostText.align = TextFormatAlign.RIGHT;
_partialCostText.constrictTextToSize
```

```
= false;
_partialCostText.x = 38;
_partialCostText.y = 30;

_partialCostText.text = String(partialCost);

_fullCostText = new Label(18, 0xf0f0f0, 50, 25, false);
_fullCostText.align = TextFormatAlign.RIGHT;
_fullCostText.constrictTextToSize = false;
_fullCostText.x = _fullPurchaseBtn.x + 40;
_fullCostText.y = 30;
_fullCostText.text = String(fullCost);

addChild(_partialPurchaseBtn);
addChild(_fullPurchaseBtn);

addChild(_cannotAffordPartialPurchaseBtn);
addChild(_cannotFullPurchaseBtn);

addChild(_partialCostText);
addChild(_fullCostText);

addChild(_leftPremiumSymbol);
addChild(_rightPremiumSymbol);

updateBasedOnHardCurrency();
}

public function set fullCost( fullCost:int ):void
{
_fullCost = fullCost;
_fullCostText.text = String(fullCost);
updateBasedOnHardCurrency();
}

public function set partialCost( partialCost:int ):void
{
_partialCost = partialCost;
_partialCostText.text = String(partialCost);
updateBasedOnHardCurrency();
}

private function updateBasedOnHardCurrency():void
{
var currentPremium:uint = CurrentUser.wallet.premium;
var canPurchasePartial:Boolean = (_partialCost <= currentPremium)
var canPurchaseFull:Boolean = (_fullCost <= currentPremium)

if(CONFIG::IS_CRYPTO && _partialCost == 0 && _fullCost == 0)
{
_leftPremiumSymbol.visible
```

```
= false;
_partialCostText.visible = false;

_partialPurchaseBtn.visible = false;
_cannotAffordPartialPurchaseBtn.visible = false;

_fullPurchaseBtn.visible = true;
_cannotFullPurchaseBtn.visible = false;
}
else
{
_leftPremiumSymbol.visible = true;
_partialCostText.visible = true;

_partialPurchaseBtn.visible = canPurchasePartial;
_cannotAffordPartialPurchaseBtn.visible = !canPurchasePartial;

_fullPurchaseBtn.visible = canPurchaseFull;
_cannotFullPurchaseBtn.visible = !canPurchaseFull;
}
}

public function set enabled( value:Boolean ):void
{
_enabled = value;



if(CONFIG::IS_CRYPTO && _partialCost == 0 && _fullCost == 0)
{
_leftPremiumSymbol.visible = false;
_partialCostText.visible = false;

partialPurchaseEnabled = false;
fullPurchaseBtnEnabled = _enabled;

cannotPartialPurchaseEnabled = false;
cannotFullPurchaseBtnEnabled = false;
}
else
{
_leftPremiumSymbol.visible = true;
_partialCostText.visible = true;

partialPurchaseEnabled = _enabled;
fullPurchaseBtnEnabled = _enabled;

cannotPartialPurchaseEnabled = _enabled;
cannotFullPurchaseBtnEnabled = _enabled;
}
if
```

```
(_enabled)
updateBasedOnHardCurrency();
}

public function get enabled():Boolean { return _enabled; }

public function set partialPurchaseEnabled( enabled:Boolean ):void
{
_partialPurchaseBtn.enabled = enabled;
_leftPremiumSymbol.visible = enabled;
_rightPremiumSymbol.visible = enabled;
_fullCostText.visible = enabled;
if (!enabled)
{
_partialPurchaseBtn.fontSize = _disabledFontSize;
_partialPurchaseBtn.text = _offlineText;
_partialPurchaseBtn.setMargin(20, 10);
_partialPurchaseBtn.label.textColor = 0x929699;
} else
{
_partialPurchaseBtn.fontSize = _enabledFontSize;
_partialPurchaseBtn.text = _partialPurchaseBtnProto.text;
_partialPurchaseBtn.label.y -= 3;
_partialPurchaseBtn.label.textColor = 0xd9ab70;
}
}

public function set fullPurchaseBtnEnabled( enabled:Boolean ):void
{
_fullPurchaseBtn.enabled = enabled;
_partialCostText.visible = enabled;
if (!enabled)
{
_fullPurchaseBtn.fontSize = _disabledFontSize;
_fullPurchaseBtn.text = _offlineText;
_fullPurchaseBtn.setMargin(15, 10);
_fullPurchaseBtn.label.textColor = 0x929699;
} else
{
_fullPurchaseBtn.fontSize = _enabledFontSize;
_fullPurchaseBtn.text = _fullPurchaseBtnProto.text;
_fullPurchaseBtn.label.y -= 3;
_fullPurchaseBtn.label.textColor = 0xd9ab70;

}
}

public function set cannotPartialPurchaseEnabled( enabled:Boolean ):void
{
_cannotAffordPartialPurchaseBtn.enabled = enabled;
if
```

```
(!enabled)
{
_cannotAffordPartialPurchaseBtn.fontSize = _disabledFontSize;
_cannotAffordPartialPurchaseBtn.text = _offlineText;
_cannotAffordPartialPurchaseBtn.setMargin(20, 10);
_cannotAffordPartialPurchaseBtn.label.textColor = 0x929699;
} else
{
_cannotAffordPartialPurchaseBtn.fontSize = _enabledFontSize;
_cannotAffordPartialPurchaseBtn.text = _cannotPartialCostBtnProto.text;
_cannotAffordPartialPurchaseBtn.label.y -= 3;
_cannotAffordPartialPurchaseBtn.label.textColor = 0xd9ab70;
}
}

public function set cannotFullPurchaseBtnEnabled( enabled:Boolean ):void
{
_cannotFullPurchaseBtn.enabled = enabled;
if (!enabled)
{
_cannotFullPurchaseBtn.fontSize = _disabledFontSize;
_cannotFullPurchaseBtn.text = _offlineText;
_cannotFullPurchaseBtn.setMargin(20, 10);
_cannotFullPurchaseBtn.label.textColor = 0x929699;
} else
{
_cannotFullPurchaseBtn.fontSize = _enabledFontSize;
_cannotFullPurchaseBtn.text = _cannotAffordFullCostBtnProto.text;
_cannotFullPurchaseBtn.label.y -= 3;
_cannotFullPurchaseBtn.label.textColor = 0xd9ab70;
}
}

public function destroy():void
{
if (_partialPurchaseBtn)
{
_partialPurchaseBtn.removeEventListener(MouseEvent.CLICK,
_partialPurchaseBtnProto.callback);
_partialPurchaseBtn.destroy();
}

_partialPurchaseBtn = null;
_partialPurchaseBtnProto = null;

if (_fullPurchaseBtn)
{
_fullPurchaseBtn.removeEventListener(MouseEvent.CLICK, _fullPurchaseBtnProto.callback);
_fullPurchaseBtn.destroy();
}

_fullPurchaseBtn
```

```
= null;
_fullPurchaseBtnProto = null;

if (_partialCostText)
_partialCostText.destroy();

_partialCostText = null;

if (_fullCostText)
_fullCostText.destroy();

_fullCostText = null;

_leftPremiumSymbol = null;
_rightPremiumSymbol = null;
}
}
}

-----------------------------------
File 699: igw\com\ui\core\component\misc\FleetIconLayoutComponent.as
package com.ui.core.component.misc
{
import com.game.entity.components.battle.Ship;
import com.model.battle.BattleEntityVO;
import com.ui.core.component.IComponent;
import com.ui.modal.PanelFactory;
import com.ui.modal.dock.ShipIcon;

import flash.display.Sprite;

public class FleetIconLayoutComponent extends Sprite implements IComponent
{
private var _selectedFleetShips:Vector.<ShipIcon>;
private var _enabled:Boolean;

public function FleetIconLayoutComponent()
{
super();

//Init Ship Icons
_selectedFleetShips = new Vector.<ShipIcon>();
var image:ShipIcon;
var xPos:uint;
var yPos:uint;
for (var i:uint = 0; i < 6; ++i)
{
image = new ShipIcon();
image.scale(0.39, 0.39);
//image.setShowBar(true);

switch
```

```
(i)
{
case 0:
xPos = 45; //94;
yPos = 0; //20;
break;
case 1:
xPos = 0; //49;
yPos = 23; //44;
break;
case 2:
xPos = 90; //139;
yPos = 23; //44;
break;
case 3:
xPos = 0; //49;
yPos = 75; //96;
break;
case 4:
xPos = 90; //139;
yPos = 75; //96;
break;
case 5:
xPos = 45; //94;
yPos = 102; //123;
break;
}

image.x = xPos;
image.y = yPos;
addChild(image);
_selectedFleetShips.push(image);
}
_enabled = true;
}

public function setUpEnemyFleetIcon( ships:Vector.<BattleEntityVO>, callback:Function ):void
{
var currentImage:ShipIcon;
var i:uint;
var len:uint = ships.length;
for (; i < len; ++i)
{
currentImage = _selectedFleetShips[i];
currentImage.clearImageBitmap();
currentImage.mouseEnabled = false;

currentImage.onLoadShipImage.add(callback);
currentImage.setShip(null, ships[i].prototype);
currentImage.setBarValue(1 - ships[i].healthPercent);
}
```

```
len = _selectedFleetShips.length;
for (i = 0; i < len; ++i)
_selectedFleetShips[i].mouseEnabled = false;
}

public function get enabled():Boolean
{
return _enabled;
}

public function set enabled( value:Boolean ):void
{
_enabled = value;
var len:uint = _selectedFleetShips.length;
var currentIcon:ShipIcon;
for (var i:uint = 0; i < len; ++i)
{
currentIcon = _selectedFleetShips[i];
currentIcon.enabled = _enabled;
}
}

public function destroy():void
{
var len:uint = _selectedFleetShips.length;
var currentIcon:ShipIcon;
for (var i:uint = 0; i < len; ++i)
{
currentIcon = _selectedFleetShips[i];
currentIcon.destroy();
currentIcon = null;
}
_selectedFleetShips.length = 0;
}
}
}

-------------------------------------
File 700: igw\com\ui\core\component\misc\ImageComponent.as
package com.ui.core.component.misc
{
import com.ui.core.component.IComponent;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;

public class ImageComponent extends Sprite implements IComponent
{
```

```
protected var _bitmap:Bitmap;
protected var _imageHeight:Number;
protected var _imageWidth:Number;
protected var _center:Boolean;

public function init( imageWidth:Number, imageHeight:Number ):void
{
_imageHeight = imageHeight;
_imageWidth = imageWidth;

if (!_bitmap)
{
_bitmap = new Bitmap();
addChild(_bitmap);
}
}

public function onImageLoaded( asset:BitmapData ):void
{
if (asset && _bitmap)
{
_bitmap.bitmapData = asset;
_bitmap.width = asset.width;
_bitmap.height = asset.height;
var scale:Number = 0;
if (_bitmap.width > _imageWidth)
{
scale = _imageWidth / _bitmap.width;
_bitmap.width *= scale;
_bitmap.height *= scale;
}
if (_bitmap.height > _imageHeight)
{
scale = _imageHeight / _bitmap.height;
_bitmap.width *= scale;
_bitmap.height *= scale;
}

if (_center)
{
_bitmap.x = (_imageWidth - _bitmap.width) * 0.5;
_bitmap.y = (_imageHeight - _bitmap.height) * 0.5;
}
}
}

public function get image():Bitmap
{
return _bitmap;
}
```

```
public function clearBitmap():void { if (_bitmap) _bitmap.bitmapData = null; }

override public function get height():Number { return (_bitmap) ? _bitmap.height : 0; }
override public function get width():Number { return (_bitmap) ? _bitmap.width : 0; }

public function get center():Boolean { return _center; }
public function set center( value:Boolean ):void { _center = value; }

public function get enabled():Boolean { return visible; }
public function set enabled( value:Boolean ):void { visible = value; }

public function get smoothing():Boolean { return (_bitmap) ? _bitmap.smoothing : 0; }
public function set smoothing( v:Boolean ):void { if (_bitmap) _bitmap.smoothing = v; }

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);

alpha = 1;
center = false;
mask = null;
filters = [];
x = y = 0;
visible = true;

if (_bitmap)
{
_bitmap.bitmapData = null;
_bitmap = null;
}
}
}
}

-------------------------------------
File 701: igw\com\ui\core\component\misc\TooltipComponent.as
package com.ui.core.component.misc
{
import com.enum.ui.LabelEnum;
import com.ui.UIFactory;
import com.ui.core.component.IComponent;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.text.TextFormatAlign;

public class TooltipComponent extends Sprite implements IComponent
{
```

```
private var _numCols:int;
private var _tipLabels:Vector.<Label>;
private var _tipStrings:Vector.<String>

public function TooltipComponent()
{
super();

_tipLabels = new Vector.<Label>;
_tipStrings = new Vector.<String>;
}

public function init( numCols:int = 1, labelWidth:int = 100, labelHeight:int = 100, fontSize:int = 12,
labelFont:String = 'Open Sans' ):void
{
_numCols = numCols;

var label:Label;
for (var i:int = 0; i < numCols; i++)
{
label = UIFactory.getLabel((labelFont == "Open Sans") ? LabelEnum.DEFAULT_OPEN_SANS :
LabelEnum.DEFAULT, labelWidth / numCols, labelHeight);
label.align = TextFormatAlign.LEFT;
label.fontSize = fontSize;
label.constrictTextToSize = false;
label.multiline = true;
_tipLabels.push(label);
}
}

public function layoutTooltip( tooltip:String, dx:int = 0, dy:int = 0, padding:int = 6, leading:int = 0
):void
{
if (tooltip)
{
var i:int = 0
var endTag:String;
var count:int = 0;
var oldIdx:int = 0;
var elementsPerCol:int = 0;

//Clear out old strings; otherwise, they just keep getting added if the window isn't closed
_tipStrings.length = 0;

for (i = 0; i <= tooltip.length; i++)
{
endTag = tooltip.substr(i, 6);

if (endTag == '<br/>\n')
{
```

```
if (oldIdx != 0)
oldIdx += 6;

_tipStrings.push(tooltip.substring(oldIdx, i + 5));

oldIdx = i;

count++;

}
}

elementsPerCol = Math.ceil(count / _numCols);
var unevenElements:Boolean = false;
//If the elements are uneven we need to account for that in the last column
if ((count % _numCols) != 0)
unevenElements = true;


for (i = 0; i < _numCols; i++)
{
//Make sure labels don't have any left over text
_tipLabels[i].text = _tipLabels[i].htmlText = '';

//Set up leading for the tooltip
_tipLabels[i].htmlText += '<textformat leading="' + leading + '">';
for (var j:int = 0; j < elementsPerCol; j++)
{
if (unevenElements && _numCols == 3 && i == (_numCols - 1) && j == (elementsPerCol - 2))
break;

if (j + elementsPerCol * i < _tipStrings.length)
Label(_tipLabels[i]).htmlText += _tipStrings[j + elementsPerCol * i];

if (unevenElements && _numCols == 2 && i == (_numCols - 1) && j == (elementsPerCol - 2))
break;

}
//Close leading tag
_tipLabels[i].htmlText += '</textformat>';

if (i != 0)
_tipLabels[i].x = dx + Label(_tipLabels[i - 1]).width * i + padding;
else
_tipLabels[i].x = dx;
_tipLabels[i].y = dy;


addChild(_tipLabels[i]);
}
```

```
}
}

public function get enabled():Boolean { return false; }
public function set enabled( value:Boolean ):void {}

public function destroy():void
{
for each (var label:Label in _tipLabels)
{
if (contains(label))
removeChild(label);
//UIFactory.destroyLabel(label);
}

_tipLabels.length = 0;
_tipStrings.length = 0;
x = y = 0;
visible = true;
}
}
}

-------------------------------------
File 702: igw\com\ui\core\component\page\Page.as
package com.ui.core.component.page
{
import flash.display.Sprite;

import org.shared.ObjectPool;

public class Page extends Sprite
{
private var _icons:Vector.<PageIcon>;
private var _component:PageComponent;

public function init( component:PageComponent, IconClass:Class ):void
{
if (!IconClass)
return;

_icons = new Vector.<PageIcon>;
_component = component;
for (var i:int = 0; i < component.iconsPerPage; i++)
{
var icon:PageIcon = ObjectPool.get(IconClass);
icon.init();
icon.x = ((i % component.iconsPerRow) * component.iconWidth) + ((i %
component.iconsPerRow) * component.iconSpacing);
var
```

```
row:int = Math.floor(i / component.iconsPerRow);
icon.y = (row * component.iconHeight) + (row * component.iconVertSpacing);
icon.visible = false;
_icons.push(icon);
addChild(icon);
}
}

public function update( list:*, pageIndex:int ):void
{
var startIndex:int = pageIndex * _component.iconsPerPage;
var end:int = startIndex + _component.iconsPerPage;
if (end > list.length)
end = list.length - startIndex;
for (var i:int = 0; i < _component.iconsPerPage; i++)
{
if (i + startIndex < list.length)
{
_icons[i].update(list[startIndex + i]);
_icons[i].visible = true;
} else
_icons[i].visible = false;
}
}

public function get icons():Vector.<PageIcon> { return _icons; }

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);

for (var i:int = 0; i < _icons.length; i++)
ObjectPool.give(_icons[i]);
_icons = null;
}
}
}

-----------------------------------
File 703: igw\com\ui\core\component\page\PageComponent.as
package com.ui.core.component.page
{
import flash.display.Sprite;
import flash.geom.Rectangle;

import org.greensock.TweenLite;
import org.shared.ObjectPool;

/**
*
```

```
@author Phillip Reagan
*/
public class PageComponent extends Sprite
{
private static const PAGE_SPACING:int = 15;
private static const FLIP_RIGHT:String = "flipRight";
private static const FLIP_LEFT:String = "flipLeft";

//set these before calling init
public var iconWidth:int = 0;
public var iconHeight:int = 0;
public var iconSpacing:int = 0;
public var iconVertSpacing:int = 0;
public var iconsPerPage:int = 0;
public var iconsPerRow:int = 0;
public var iconClass:Class;

private var _page1:Page;
private var _page2:Page;
private var _visiblePage:Page;
private var _scrollRect:Rectangle;
private var _animating:Boolean = false;
private var _data:*;
private var _pageIndex:int;
private var _nextMove:String;

public function init():void
{
_page1 = ObjectPool.get(Page);
_page1.init(this, iconClass);
_page2 = ObjectPool.get(Page);
_page2.init(this, iconClass);
_page2.x = _page1.width + PAGE_SPACING;
_visiblePage = _page1;

addChild(_page1);
addChild(_page2);

_scrollRect = new Rectangle(0, 0, _page1.width, _page1.height);
scrollRect = _scrollRect;
}

public function format( icon_width:int,
icon_height:int,
icon_spacing:int,
icon_VertSpacing:int,
icons_PerPage:int,
icons_PerRow:int,
icon_class:Class ):void
{
iconWidth
```

```
= icon_width;
iconHeight = icon_height;
iconSpacing = icon_spacing;
iconVertSpacing = icon_VertSpacing;
iconsPerPage = icons_PerPage;
iconsPerRow = icons_PerRow;
iconClass = icon_class;
}

//got new data so reset and show it starting from the supplied pageIndex
public function update( data:*, pageIndex:int = 0 ):void
{
try
{
var test:int = data.length;
} catch ( e:Error )
{
return;
}

_data = data;
_pageIndex = pageIndex;
TweenLite.killTweensOf(_page1);
TweenLite.killTweensOf(_page2);
_page1.x = 0;
_page2.x = _page1.width + PAGE_SPACING;
_visiblePage = _page1;
_animating = false;
flipToPage(_pageIndex);
}

public function flipToPage( pageIndex:int ):void
{
if (_pageIndex == pageIndex)
_visiblePage.update(_data, _pageIndex);
else
{
var goRight:Boolean = _pageIndex < pageIndex;
_pageIndex = pageIndex
if (goRight)
flipRight();
else
flipLeft();
}
}

public function gotoItem( item:* ):*
{
if (_data.length)
{
for
```

```
(var i:int = 0; i < _data.length; i++)
{
if (_data[i] == item)
{
var pageIndex:int = Math.floor(i / iconsPerPage);
var iconIndex:int = i - (pageIndex * iconsPerPage);
flipToPage(pageIndex);
return _visiblePage.icons[iconIndex];
}
}
}
return null;
}

private function flipRight():void
{
if (_animating)
_nextMove = FLIP_RIGHT;
else
{
var targetPage:Page = (_visiblePage == _page1) ? _page2 : _page1;
if (targetPage.x < _visiblePage.x)
targetPage.x = _visiblePage.width + PAGE_SPACING;
targetPage.update(_data, _pageIndex);
TweenLite.to(_visiblePage, .7, {x:-targetPage.x});
TweenLite.to(targetPage, .7, {x:0, onComplete:animationComplete});
_visiblePage = targetPage;
_animating = true;
}
}

private function flipLeft():void
{
if (_animating)
_nextMove = FLIP_LEFT;
else
{
var targetPage:Page = (_visiblePage == _page1) ? _page2 : _page1;
if (targetPage.x > _visiblePage.x)
targetPage.x = -targetPage.x;
targetPage.update(_data, _pageIndex);
TweenLite.to(_visiblePage, .7, {x:_visiblePage.width + PAGE_SPACING});
TweenLite.to(targetPage, .7, {x:0, onComplete:animationComplete});
_visiblePage = targetPage;
_animating = true;
}
}

private function animationComplete():void
{
_animating
```

```
= false;
if (_nextMove)
{
if (_nextMove == FLIP_RIGHT)
flipRight();
else
flipLeft();
_nextMove = null;
}
}

public function get page1():Page { return _page1; }
public function get page2():Page { return _page2; }

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);

TweenLite.killTweensOf(_page1);
TweenLite.killTweensOf(_page2);

ObjectPool.give(_page1);
ObjectPool.give(_page2);

_page1 = null;
_page2 = null;
_visiblePage = null;
_scrollRect = null;
_data = null;
}
}
}

-----------------------------------
File 704: igw\com\ui\core\component\page\PageEvent.as
package com.ui.core.component.page
{
import flash.events.Event;

public class PageEvent extends Event
{
public static const ITEM_SELECTED:String = 'ItemSelected';

public var selection:*;

public function PageEvent( selection:* )
{
super(ITEM_SELECTED, true, true);
this.selection = selection;
}
```

```
}
}

-----------------------------------
File 705: igw\com\ui\core\component\page\PageIcon.as
package com.ui.core.component.page
{
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Sprite;
import flash.events.MouseEvent;

public class PageIcon extends Sprite
{
protected var _onRollOverSound:String = 'sounds/sfx/AFX_UI_Mouse2_V001A.mp3';
protected var _onClickSound:String = 'sounds/sfx/AFX_UI_Mouse1_V001A.mp3';
protected var _enabled:Boolean;

public function init():void
{
enabled = true;
}

public function update( vo:* ):void
{

}

public function destroy():void
{

}

public function set enabled( enabled:Boolean ):void
{
_enabled = enabled;
}

protected function onMouseClick( e:MouseEvent ):void
{
if (_onClickSound != '')
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_1, 0.5);
}

protected function onMouseRollOver( e:MouseEvent ):void
{
if (_onRollOverSound != '')
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_2,
```

```
0.5);
}
}
}

-----------------------------------
File 706: igw\com\ui\core\component\pips\PipComponent.as
package com.ui.core.component.pips
{
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.utils.getDefinitionByName;

/**
* The PipComponent provides a convenient way to add pip navigation to the interface. Set
'totalPips'
* equal to the number of pips you want to show. Set the initially selected pip by setting 'selected'
* equal to the desired pip index. Clicking on a pip will dispatch a PipEvent to notify which pip
was selected.
* Change the size and spacing of the pips via 'pipSize' and 'spacing'.
*
* @author Phillip Reagan
* @langversion ActionScript 3.0
* @playerversion Flash 10.0
*/
public class PipComponent extends Sprite
{
private static const PIP_UP_COLOR:uint = 0x111111;
private static const PIP_DOWN_COLOR:uint = 0x0F8584;
private static const PIP_OVER_COLOR:uint = 0x23A4A2;
private static const PIP_SELECTED_COLOR:uint = 0x17DBDB;

private var _pipUpBMD:BitmapData;
private var _pipDownBMD:BitmapData;
private var _pipOverBMD:BitmapData;
private var _pipSelectedBMD:BitmapData;
private var _pipUnreadBMD:BitmapData;
private var _pips:Array;
private var _pooledPips:Array; //keep a list of unused pips in case we need to use them again
private var _showPips:Boolean;
private var _showArrows:Boolean;
private var _arrowLeft:BitmapButton;
private var _arrowRight:BitmapButton;
private
```

```
var _pipUnreadClass:Class;

private var _totalPips:int;
public var spacing:Number = 4;
public var pipSize:Number = 8;

public function init( showPips:Boolean, showArrows:Boolean ):void
{
_showPips = showPips;
_showArrows = showArrows;

var pipUnreadClass:Class = Class(getDefinitionByName(('MotDPipNewBMD')));
_pipUnreadBMD = BitmapData(new pipUnreadClass());

_arrowLeft = UIFactory.getButton(ButtonEnum.BACK_ARROW);
_arrowLeft.addEventListener(MouseEvent.CLICK, onShiftLeft, false, 0, true);
_arrowLeft.visible = _showArrows;

_arrowRight = UIFactory.getButton(ButtonEnum.FORWARD_ARROW);
_arrowRight.x = _arrowLeft.x + _arrowLeft.width + spacing;
_arrowRight.y = 0;
_arrowRight.addEventListener(MouseEvent.CLICK, onShiftRight, false, 0, true);
_arrowRight.visible = _showArrows;

addChild(_arrowLeft);
addChild(_arrowRight);
}

private function addPip():void
{
var pip:BitmapButton = _pooledPips.shift();
if (!pip)
{
pip = new BitmapButton();
pip.init(_pipUpBMD, _pipOverBMD, _pipDownBMD, _pipDownBMD, _pipSelectedBMD);
pip.addEventListener(MouseEvent.CLICK, onPipClick, false, 0, true);
}

pip.visible = _showPips;
_pips.push(pip);
addChild(pip);
}

private function removePip():void
{
var pip:BitmapButton = _pips.shift();
removeChild(pip);
_pooledPips.push(pip);
}

private
```

```
function layout():void
{
var len:uint = _pips.length;
var currentPip:BitmapButton;

var xPos:int;
if (_showArrows)
xPos = _arrowLeft.width + spacing;

for (var i:uint = 0; i < len; ++i)
{
currentPip = _pips[i];
currentPip.x = xPos;
if (_showArrows)
currentPip.y = _arrowRight.y + (_arrowRight.height - currentPip.height) * 0.5;

xPos += currentPip.width + spacing;
}

_arrowRight.x = xPos;
}

private function onPipClick( e:MouseEvent ):void
{
var index:int = _pips.indexOf(e.currentTarget);
if (index != -1 && index != selected)
{
var oldIndex:int = selected;
selected = index;
dispatchEvent(new PipEvent(PipEvent.PIP_CLICKED, index, oldIndex));
updateArrowState();
}
}

private function onShiftLeft( e:MouseEvent ):void
{
var oldIndex:int = selected;
--selected;
dispatchEvent(new PipEvent(PipEvent.PIP_CLICKED, selected, oldIndex))
updateArrowState();
}

private function onShiftRight( e:MouseEvent ):void
{
var oldIndex:int = selected;
++selected;
dispatchEvent(new PipEvent(PipEvent.PIP_CLICKED, selected, oldIndex))
updateArrowState();
}

private
```

```
function updateArrowState():void
{
var selectedValue:int = selected;

if (selectedValue == -1)
{
_arrowRight.enabled = false;
_arrowLeft.enabled = false;
} else
{
if (selectedValue != (_totalPips - 1))
{
if (!_arrowRight.enabled)
_arrowRight.enabled = true;
} else
{
if (_arrowRight.enabled)
_arrowRight.enabled = false;
}

if (selectedValue != 0)
{
if (!_arrowLeft.enabled)
_arrowLeft.enabled = true;
} else
{
if (_arrowLeft.enabled)
_arrowLeft.enabled = false;
}
}
}

private function createPipBitmapData():void
{
// var pip:Sprite = new Sprite();
// _pipUpBMD = drawPip(pip, PIP_UP_COLOR);
// _pipDownBMD = drawPip(pip, PIP_DOWN_COLOR);
// _pipOverBMD = drawPip(pip, PIP_OVER_COLOR);
// _pipSelectedBMD = drawPip(pip, PIP_SELECTED_COLOR);
var pipUp:Class = Class(getDefinitionByName(('MotDPipNeutralBMD')));
var pipDown:Class = Class(getDefinitionByName(('MotDPipSelectedBMD')));
// var pipSelected:Class = Class(getDefinitionByName(('MotDPipNeutralBMD')));

_pipUpBMD = BitmapData(new pipUp());
_pipDownBMD = BitmapData(new pipDown());
_pipOverBMD = BitmapData(new pipDown());
_pipSelectedBMD = BitmapData(new pipDown());
_pips = [];
_pooledPips = [];
}

//
```

```
private function drawPip( pip:Sprite, col:uint ):BitmapData
// {
// pip.graphics.clear();
// pip.graphics.beginFill(col);
// pip.graphics.drawCircle(pipSize, pipSize, pipSize);
// pip.graphics.endFill();
//
// var bmd:BitmapData = new BitmapData(pip.width, pip.height, true, 0);
// bmd.draw(pip, null, null, null, null, true);
// return bmd;
// }

public function get totalPips():int { return _totalPips; }
public function set totalPips( value:int ):void
{
_totalPips = value;
if (value < 0)
return;
if (!_pipUpBMD)
createPipBitmapData();
if (value == 1)
value = 0;
while (_pips.length > value)
removePip();
//reposition
var pip:BitmapButton;
for (var i:int = 0; i < _pips.length; i++)
{
pip = _pips[i];
pip.x = (i != 0) ? _pips[i - 1].x + pip.width + spacing : 0;
}
while (_pips.length < value)
addPip();

updateArrowState();
layout();
}

public function get selected():int
{
for (var i:int = 0; i < _pips.length; i++)
{
if (_pips[i].selected)
return i;
}
return -1;
}
public function set selected( v:int ):void
{
for (var i:int = 0; i < _pips.length; i++)
_pips[i].selected
```

```
= i == v;

updateArrowState();
}

public function setPipState( idx:uint, isRead:Boolean ):void
{
if (isRead)
BitmapButton(_pips[idx]).updateBackgrounds(_pipUpBMD);
else
BitmapButton(_pips[idx]).updateBackgrounds(_pipUnreadBMD);
}

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);

if (_pips)
{
for (var i:int = 0; i < _pips.length; i++)
_pips[i].destroy();
for (i = 0; i < _pooledPips.length; i++)
_pooledPips[i].destroy();
}
_pips = null;
_pooledPips = null;
_pipUpBMD = null;
_pipDownBMD = null;
_pipOverBMD = null;
_pipSelectedBMD = null;
}
}
}

-----------------------------------
File 707: igw\com\ui\core\component\pips\PipEvent.as
package com.ui.core.component.pips
{
import flash.events.Event;

public class PipEvent extends Event
{
public static const PIP_CLICKED:String = "PageChanged";

public var oldIndex:int;
public var index:int;

public function PipEvent( type:String, pipIndex:int, pipOldIndex:int )
{
super(type,
```

```
true, true);
index = pipIndex;
oldIndex = pipOldIndex;
}
}
}
```

File 708: igw\com\ui\core\component\pulldown\DrawerComponent.as

```
package com.ui.core.component.pulldown
{
import com.Application;
import com.enum.ui.ButtonEnum;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Point;
import flash.geom.Rectangle;

import org.osflash.signals.Signal;

/**
*
*
*/
public class DrawerComponent extends Sprite implements IComponent
{
public static const EXPANDS_DOWN:int = 0;
public static const EXPANDS_LEFT:int = 1;
public static const EXPANDS_RIGHT:int = 2;
public static const EXPANDS_UP:int = 3;

private static const SPEED:int = 20;

protected var _arrow:Bitmap;

protected var _clickPos:Point;

protected var _elementHolder:Sprite;
protected var _holder:Sprite;

protected var _elements:Vector.<*>;

protected var _canDragExpand:Boolean = true;
protected
```

```
var _dragging:Boolean;
protected var _enabled:Boolean;
protected var _expanded:Boolean;
protected var _expanding:Boolean;
protected var _clickedOn:Boolean;

protected var _expansionDirection:int;
protected var _maxExpansion:int;

protected var _innerPanel:ScaleBitmap;
protected var _panel:ScaleBitmap;

protected var _marginTab:Number;
protected var _marginTail:Number;
protected var _originalHeight:Number;

protected var _scrollRect:Rectangle;

protected var _tab:BitmapButton;

protected var _updateSignal:Signal;

/**
*
* @param panelType
* @param panelSize
* @param tabSize
* @param expansionDirection
* @param marginTab
* @param marginTail
*/
public function init( panelType:String, panelSize:int, tabSize:int, expansionDirection:int,
marginTab:Number = 10, marginTail:Number = 10 ):void
{
_clickPos = new Point();
_expansionDirection = expansionDirection;
_marginTab = marginTab;
_marginTail = marginTail;
_maxExpansion = 0;
_updateSignal = new Signal();

_holder = new Sprite();
_panel = UIFactory.getPanel(panelType, panelSize, 0);

//create the tab
_tab = UIFactory.getButton(ButtonEnum.DROP_TAB, tabSize, 0);
_arrow = UIFactory.getBitmap("ArrowBMD");
_arrow.x = (tabSize - _arrow.width) * .5;
_tab.x = (panelSize - tabSize) * .5 + .5;
_tab.y = _panel.height - 1;

//elements
```

```
_elementHolder = new Sprite();
_elements = new Vector.<*>;

//orient
switch (_expansionDirection)
{
case EXPANDS_DOWN:
break;
case EXPANDS_LEFT:
_holder.rotation = 90;
break;
case EXPANDS_RIGHT:
_holder.rotation = -90;
_holder.y += panelSize;
break;
case EXPANDS_UP:
_holder.scaleY = -1;
break;
}

_dragging = _expanding = false;
expanded = false;
enabled = true;

//add the elements
_tab.addChild(_arrow);
_holder.addChild(_panel);
_holder.addChild(_tab);
addChild(_holder);
addChild(_elementHolder);
}

public function addElement( element:* ):void
{
var index:int = _elements.indexOf(element);
if (index == -1)
{
_elements.push(element);
_elementHolder.addChild(element);
findMaxExpansion();
positionHolder();
if (_panel.height > _panel.src.height)
{
maximize(false);
}
}
}

public function removeAllElements():void
{
```

```
while (_elements.length > 0)
{
removeChild(_elements[0]);
_elements.shift();
}
_maxExpansion = 0;
positionHolder();
}

public function removeElement( element:* ):void
{
var index:int = _elements.indexOf(element);
if (index != -1)
{
_elements.splice(index, 1);
_elementHolder.removeChild(element);
positionHolder();
}
findMaxExpansion();
}

protected function onMouseDown( e:MouseEvent ):void
{
removeEventListener(Event.ENTER_FRAME, onUpdate);
_clickedOn = true;
_clickPos.setTo(Application.STAGE.mouseX, Application.STAGE.mouseY);
_originalHeight = _panel.height;
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
Application.STAGE.addEventListener(MouseEvent.MOUSE_MOVE, onMouseMove, false, 0,
true);
}

protected function onMouseUp( e:MouseEvent ):void
{
if (e.type == MouseEvent.RELEASE_OUTSIDE && !_canDragExpand)
return;

if (!_clickedOn)
return;

_clickedOn = false;
removeEventListener(Event.ENTER_FRAME, onUpdate);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
if (_dragging)
{
_dragging = false;
} else if (!hasEventListener(Event.ENTER_FRAME) && _maxExpansion > 0)
{
if (!_expanded)
{
```

```
expanded = !_expanded;
_expanding = _expanded;
} else
_expanding = !_expanded;
addEventListener(Event.ENTER_FRAME, onUpdate, false, 0, true);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
}
}

protected function onMouseMove( e:MouseEvent ):void
{
if (!_canDragExpand)
return;
var panelHeight:Number = 0;
switch (_expansionDirection)
{
case EXPANDS_DOWN:
panelHeight = Application.STAGE.mouseY - _clickPos.y;
break;
case EXPANDS_LEFT:
panelHeight = _clickPos.x - Application.STAGE.mouseX;
break;
case EXPANDS_RIGHT:
panelHeight = Application.STAGE.mouseX - _clickPos.x;
break;
case EXPANDS_UP:
panelHeight = _clickPos.y - Application.STAGE.mouseY;
break;
}
if (Math.abs(panelHeight) > 5)
_dragging = true;
panelHeight = _originalHeight + panelHeight;
panelHeight = panelHeight < 0 ? 0 : panelHeight > _maxExpansion ? _maxExpansion :
panelHeight;
expanded = panelHeight > _panel.src.height;
_panel.height = panelHeight;
if (_innerPanel)
_innerPanel.height = _panel.height - 4;
_tab.y = _panel.height - 1;
positionHolder();
_updateSignal.dispatch();
}

/**
*
*
*/
protected function onUpdate( e:Event ):void
{
if
```

```
(_expanding)
{
_panel.height += SPEED;
if (_panel.height >= _maxExpansion)
{
_panel.height = _maxExpansion;
removeEventListener(Event.ENTER_FRAME, onUpdate);
}
} else
{
_panel.height -= SPEED;
if (_panel.height <= _panel.src.height)
{
_panel.height = 0;
removeEventListener(Event.ENTER_FRAME, onUpdate);
expanded = !_expanded;
}
}
if (_innerPanel)
_innerPanel.height = _panel.height - 4;
_tab.y = _panel.height - 1;
positionHolder();
_updateSignal.dispatch();
}

/**
*
*
*/
protected function positionHolder():void
{
switch (_expansionDirection)
{
case EXPANDS_DOWN:
_elementHolder.y = (_scrollRect) ? 0 : _panel.height - _elementHolder.height - _marginTab;
break;
case EXPANDS_LEFT:
_elementHolder.x = -_panel.height + _marginTab;
break;
case EXPANDS_RIGHT:
_elementHolder.x = (_scrollRect) ? 0 : _panel.height - int(_elementHolder.width) - _marginTab;
break;
case EXPANDS_UP:
_elementHolder.y = -_panel.height + _marginTab;
break;
}
if (_scrollRect)
useMask = true;
}

/**
```

```
*
*
*/
public function maximize( instant:Boolean = true ):void
{
_expanding = true;
expanded = true;
if (instant)
_panel.height = _maxExpansion;
else
addEventListener(Event.ENTER_FRAME, onUpdate, false, 0, true);
onUpdate(null);
}

/**
*
*
*/
public function minimize( instant:Boolean = true ):void
{
_expanding = false;
expanded = false;
if (instant)
_panel.height = 0;
else
addEventListener(Event.ENTER_FRAME, onUpdate, false, 0, true);
onUpdate(null);
}

/**
*
*
*/
protected function findMaxExpansion():void
{
if (!_scrollRect)
_maxExpansion = ((_expansionDirection == EXPANDS_DOWN || _expansionDirection ==
EXPANDS_UP) ? _elementHolder.height : _elementHolder.width) + _marginTab + _marginTail;
else
{
var maxSize:Number = 0;
for (var i:int = 0; i < _elements.length; i++)
{
if (_expansionDirection == EXPANDS_DOWN || _expansionDirection == EXPANDS_UP)
{
if (_elements[i].y + _elements[i].height > maxSize)
maxSize = _elements[i].y + _elements[i].height;
} else
{
if
```

```
(_elements[i].x + _elements[i].width > maxSize)
maxSize = _elements[i].x + _elements[i].width;
}
}
_maxExpansion = (_elements.length > 0) ? maxSize + _marginTab + _marginTail : 0;
}
}

public function addUpdateListener( listener:Function ):void { _updateSignal.add(listener); }
public function removeUpdateListener( listener:Function ):void { _updateSignal.remove(listener);
}

public function get canDragExpand():Boolean { return _canDragExpand; }
public function set canDragExpand( v:Boolean ):void { _canDragExpand = v; }

public function set dirty( v:Boolean ):void { positionHolder(); findMaxExpansion(); }

public function get enabled():Boolean { return _enabled; }
public function set enabled( value:Boolean ):void
{
_tab.removeEventListener(MouseEvent.MOUSE_DOWN, onMouseDown);
_tab.removeEventListener(MouseEvent.MOUSE_UP, onMouseUp);
_tab.removeEventListener(MouseEvent.RELEASE_OUTSIDE, onMouseUp);
_enabled = value;
if (_enabled)
{
_tab.addEventListener(MouseEvent.MOUSE_DOWN, onMouseDown, false, 0, true);
_tab.addEventListener(MouseEvent.MOUSE_UP, onMouseUp, false, 0, true);
_tab.addEventListener(MouseEvent.RELEASE_OUTSIDE, onMouseUp, false, 0, true);
}
}

protected function set expanded( value:Boolean ):void
{
_expanded = value;
_arrow.scaleY = (_expanded) ? 1 : -1;
_arrow.y = (_expanded) ? 4 : 14;

if (!_expanded)
{
_panel.visible = false;
_elementHolder.visible = false;
if (_innerPanel)
_innerPanel.visible = false;
} else if (_expanded && !_panel.visible)
{
_panel.visible = true;
_elementHolder.visible = true;
if (_innerPanel)
_innerPanel.visible = true;
}
```

```
}

public function set marginTab( v:Number ):void { _marginTab = v; ; }
public function set marginTail( v:Number ):void { _marginTail = v; ; }

public function set showInnerPanel( value:Boolean ):void
{
if (value && !_innerPanel)
{
_innerPanel = UIFactory.getPanel(PanelEnum.CONTAINER_DOUBLE_NOTCHED,
_panel.width - 10, _panel.height - 4, 5);
_holder.addChild(_innerPanel);
} else if (!value && _innerPanel)
{
_holder.removeChild(_innerPanel);
_innerPanel = UIFactory.destroyPanel(_innerPanel);
}
expanded = _expanded;
}

public function set useMask( v:Boolean ):void
{
if (!_scrollRect)
_scrollRect = new Rectangle();
switch (_expansionDirection)
{
case EXPANDS_DOWN:
_scrollRect.setTo(0, (_maxExpansion - _marginTab - _marginTail) - (_panel.height -
_marginTab), _panel.width, _panel.height + _marginTab);
break;
case EXPANDS_LEFT:
_scrollRect.setTo(0, 0, _panel.height - _marginTab, _panel.width);
break;
case EXPANDS_RIGHT:
_scrollRect.setTo((_maxExpansion - _marginTab - _marginTail) - (_panel.height - _marginTab),
0, _panel.height + _marginTab, _panel.width);
break;
case EXPANDS_UP:
_scrollRect.setTo(0, 0, _panel.width, _panel.height - _marginTab);
break;
}
_elementHolder.scrollRect = _scrollRect;
}

override public function get width():Number { return (_expansionDirection == EXPANDS_DOWN
|| _expansionDirection == EXPANDS_UP) ? _panel.width : _panel.height; }
override public function get height():Number { return (_expansionDirection ==
EXPANDS_DOWN || _expansionDirection == EXPANDS_UP) ? _panel.height : _panel.width; }

public
```

```
function destroy():void
{
while (numChildren > 0)
removeChildAt(0);
removeEventListener(Event.ENTER_FRAME, onUpdate);
Application.STAGE.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
_dragging = false;
enabled = false;
showInnerPanel = false;

_arrow = null;
_clickPos = null;
_elementHolder = UIFactory.destroyPanel(_elementHolder);
_elements.length = 0;
_elements = null;
_holder = UIFactory.destroyPanel(_holder);
_panel = UIFactory.destroyPanel(_panel);
_scrollRect = null;
_tab = UIFactory.destroyButton(_tab);

_updateSignal.removeAll();
_updateSignal = null;
}
}
}
```

File 709: igw\com\ui\core\component\pulldown\PullDown.as

```
package com.ui.core.component.pulldown
{
import com.ui.core.component.IComponent;
import com.ui.modal.PanelFactory;
import com.ui.core.ScaleBitmap;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.osflash.signals.Signal;

public class PullDown extends Sprite implements IComponent
{
private var _bg:ScaleBitmap;
private var _selections:Array;
private var _selected:PullDownComponent;
private var _isExpanded:Boolean;
private var _isEnabled:Boolean;
private
```

```
var _growDown:Boolean;
private var _defaultSelectedIndex:int;
private var _defaultXPos:int;
private var _defaultYPos:int;
private var _width:Number;
private var _height:Number;
private var _fontSize:int;
private var _padding:Number;
public var onChangedSelected:Signal;

private var _nothingSelected:String = 'CodeString.PullDown.NothingSelected'; // Please Select

public function init( width:Number, height:Number, padding:Number, pullDownBG:String,
rect:Rectangle, xPos:Number = 0, yPos:Number = 0, defaultSelectedIndex:int = -1, fontSize:int
= 14, growDown:Boolean =
true ):void
{
_selections = new Array()
_defaultXPos = x = xPos;
_defaultYPos = y = yPos;

_width = width;
_height = height;
_growDown = growDown;
_fontSize = fontSize;

_defaultSelectedIndex = defaultSelectedIndex;

_padding = padding;

onChangedSelected = new Signal(Array);
_bg = PanelFactory.getScaleBitmapPanel(pullDownBG, width, height, rect);

_selected = new PullDownComponent(_width, _height, _fontSize);

_bg.width = _selected.width + _padding;
_bg.height = _selected.height + _padding;
_bg.x = -10;
_bg.y = 0;

_selected.x = _bg.x + (_bg.width - _selected.width) * 0.5
_selected.y = _bg.y + (_bg.height - _selected.height) * 0.5;

addChildAt(_bg, 0);
addChild(_selected);
enabled = true;
}

private function onSelectedClick( e:MouseEvent ):void
{
if
```

```
(!_isExpanded)
expandPullDown();
else
contractPullDown();
}

private function onRollOut( e:MouseEvent ):void
{
if (_isExpanded)
contractPullDown();
}

private function onClicked( e:MouseEvent ):void
{
var clickedPullDownComponent:PullDownComponent;
if (e.target is PullDownComponent)
{
clickedPullDownComponent = PullDownComponent(e.target);
} else if (e.target.parent is PullDownComponent)
{
clickedPullDownComponent = (e.target.parent);
}

if (clickedPullDownComponent && clickedPullDownComponent != _selected)
{
select(clickedPullDownComponent);
}
}

private function select( selectedComponent:PullDownComponent ):void
{
contractPullDown();

var dataHolder:PullDownData = _selected.data;
_selected.data = selectedComponent.data

if (dataHolder != null)
selectedComponent.data = dataHolder;
else
{
var index:uint = _selections.indexOf(selectedComponent, 0);
if (index != -1)
_selections.splice(index, 1);
}

onChangedSelected.dispatch(_selected.data.returnParams);
_selections.sortOn('index', Array.NUMERIC);
}

public function selectByIndex( index:int ):void
{
```

```
var len:uint = _selections.length;
var currentSelection:PullDownComponent;
for (var i:uint = 0; i < len; ++i)
{
currentSelection = _selections[i];
if (currentSelection.index == index)
{
select(currentSelection);
break;
}
}
}

public function selectByDisplayName( displayName:String ):void
{
var len:uint = _selections.length;
var currentSelection:PullDownComponent;
for (var i:uint = 0; i < len; ++i)
{
currentSelection = _selections[i];
if (currentSelection.displayName == displayName)
{
select(currentSelection);
break;
}
}
}

private function expandPullDown():void
{

_isExpanded = true;
var len:uint = _selections.length;
var newY:Number = _selected.y;
var currentSelection:PullDownComponent;
if (len < 1)
return;

for (var i:uint = 0; i < len; ++i)
{
currentSelection = _selections[i];
if (_growDown)
newY += _height;
else
newY -= _height;

TweenLite.to(currentSelection, .5, {y:newY, ease:Quad.easeOut});
currentSelection.visible = true;
}

var
```

```
bgHeight:int;
if (!_growDown)
{
_bg.y = newY - 10;
bgHeight = Math.abs(_bg.y) + _selected.y + _selected.height + _padding * 0.5;
} else
bgHeight = newY + _height + _padding;

_bg.height = bgHeight;
}

private function contractPullDown():void
{
_isExpanded = false;
var len:uint = _selections.length;
var currentSelection:PullDownComponent;
var newY:Number = _selected.y;
for (var i:int = (len - 1); i >= 0; --i)
{
currentSelection = _selections[i];
currentSelection.visible = false;
TweenLite.to(currentSelection, .2, {y:newY, ease:Quad.easeIn});
}

_bg.height = _selected.height + _padding;
_bg.y = 0;
}

public function addPullDownData( data:Array ):void
{
var len:uint = data.length;
var currentData:PullDownData;
var selection:PullDownComponent;

if (_defaultSelectedIndex == -1)
_selected.displayName = _nothingSelected;

for (var i:uint = 0; i < _selections.length; ++i)
{
removeChild(_selections[i]);
_selections[i].destroy();
}
_selections.length = 0;

for (i = 0; i < len; ++i)
{
currentData = data[i];
if (_defaultSelectedIndex == currentData.index)
{
_selected.data = currentData;
onChangedSelected.dispatch(_selected.data.returnParams);
```

```
} else
{
selection = new PullDownComponent(_width, _height, _fontSize);
selection.data = currentData;
selection.x = _selected.x;
selection.y = _selected.y;
selection.visible = false;
addChild(selection);
_selections.push(selection)
}
}
_selections.sortOn('index', Array.NUMERIC);
}

override public function get height():Number
{
return _height;
}

override public function get width():Number
{
return _width;
}


public function get enabled():Boolean
{
return _isEnabled;
}

private function addListeners():void
{
_selected.addEventListener(MouseEvent.CLICK, onSelectedClick, false, 0, true);
addEventListener(MouseEvent.ROLL_OUT, onRollOut, false, 0, true);
addEventListener(MouseEvent.CLICK, onClicked, false, 0, true);
}

private function removeListeners():void
{
_selected.removeEventListener(MouseEvent.CLICK, onSelectedClick);
removeEventListener(MouseEvent.ROLL_OUT, onRollOut);
removeEventListener(MouseEvent.CLICK, onClicked);
}

public function set enabled( value:Boolean ):void
{
if (_isEnabled != value)
{
_isEnabled = value;

if
```

```
(_isEnabled)
addListeners();
else
{
removeListeners();
if (_isExpanded)
contractPullDown();
}
}
}

public function set defaultSelectedIndex( defaultSelectedIndex:uint ):void
{
_defaultSelectedIndex = defaultSelectedIndex;
}

public function destroy():void
{
if (_isEnabled)
removeListeners();

while (numChildren > 0)
removeChildAt(0);

_selected.destroy();
_selected = null;

onChangedSelected.removeAll();
onChangedSelected = null;

var len:uint = _selections.length;
var currentSelection:PullDownComponent;
for (var i:uint = 0; i < len; ++i)
{
currentSelection = _selections[i];
currentSelection.destroy();
currentSelection = null;
}
_selections = null;
}
}
}

-----------------------------------
File 710: igw\com\ui\core\component\pulldown\PullDownComponent.as
package com.ui.core.component.pulldown
{
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import
```

```
flash.events.MouseEvent;
import flash.text.TextFormatAlign;

public class PullDownComponent extends Sprite
{
private var _displayName:Label;
private var _data:PullDownData;

public function PullDownComponent( width:Number, height:Number, fontSize:int )
{
_displayName = new Label(fontSize, 0x94beda, width, height);
_displayName.align = TextFormatAlign.CENTER;
_displayName.y += 2;
addEventListener(MouseEvent.ROLL_OUT, onRollOut, false, 0, true);
addEventListener(MouseEvent.ROLL_OVER, onRollOver, false, 0, true);


addChild(_displayName);
}

public function set data( data:PullDownData ):void
{
_data = data;
_displayName.textColor = _data.fontColor;
_displayName.text = _data.displayName;
}

public function set displayName( displayName:String ):void
{
_displayName.text = displayName;
}

private function onRollOut( e:MouseEvent ):void
{
if (_data != null)
_displayName.textColor = _data.fontColor;
}

private function onRollOver( e:MouseEvent ):void
{
_displayName.textColor = 0xc9e6f6;
}

public function get displayName():String { return _displayName.text; }

public function set fontSize( fontSize:int ):void { _displayName.fontSize = fontSize; }

public function get data():PullDownData { return _data; }

public function get index():uint { return _data.index; }

public
```

```
function destroy():void
{
removeEventListener(MouseEvent.ROLL_OUT, onRollOut);
removeEventListener(MouseEvent.ROLL_OVER, onRollOver);
_displayName.destroy();
_displayName = null;
_data = null;
}
}
}

------------------------------------
File 711: igw\com\ui\core\component\pulldown\PullDownData.as
package com.ui.core.component.pulldown
{
public class PullDownData
{
private var _displayName:String;
private var _index:uint;
private var _fontColor:uint;
private var _returnParams:Array;
public function PullDownData()
{
_fontColor = 0xffffff;
_returnParams = new Array();
}

public function set returnParams( returnParams:Array ):void {_returnParams = returnParams;}
public function get returnParams():Array {return _returnParams;}
public function set fontColor( fontColor:uint ):void {_fontColor = fontColor;}
public function get fontColor():uint {return _fontColor;}
public function set index( index:uint ):void {_index = index;}
public function get index():uint {return _index;}
public function set displayName( displayName:String ):void {_displayName = displayName;}
public function get displayName():String {return _displayName;}

}
}
------------------------------------
File 712: igw\com\ui\core\component\tab\TabComponent.as
package com.ui.core.component.tab
{
import com.ui.UIFactory;
import com.ui.core.component.IComponent;
import com.ui.core.component.button.BitmapButton;

import flash.display.Bitmap;
import flash.display.DisplayObject;
import flash.display.Sprite;
import flash.events.MouseEvent;
import
```

```
flash.utils.Dictionary;

import org.osflash.signals.Signal;

public class TabComponent extends Sprite implements IComponent
{
private var _automaticLayoutHorizontal:Boolean = false;
private var _automaticLayoutVertical:Boolean = false;
private var _panel:Sprite;
private var _switchTabSignal:Signal = new Signal(String);
private var _tabLookup:Dictionary;
private var _tabs:Vector.<BitmapButton> = new Vector.<BitmapButton>;

public function init( panelType:String, headerType:String, width:Number = 0, height:Number = 0,
headerSize:Number = 0 ):void
{
if (panelType && headerType)
{
_panel = UIFactory.getHeaderPanel(panelType, headerType, width, height, headerSize);
addChild(_panel);
}
_switchTabSignal = new Signal(String);

_tabLookup = new Dictionary(false);
}

/**
* Adds a tab, with a text label, to the component
*
* @param name Name is the value that is returned when the user switches tabs. It is also used
to get the tab element from the tab component.
* @param type The type of tab button to create. Any button in ButtonEnum can be used as a tab
button
* @param width Width to resize the button to
* @param height Height to resize the button to
* @param x X position of the tab in the component
* @param y Y position of the tab in the component
* @param text The text to display in the tab
* @param labelType The text format of the label in the tab
*/
public function addTab( name:String, type:String, width:int = 0, height:int = 0, x:int = 0, y:int = 0,
text:String = null, labelType:String = null ):void
{
if (!_tabLookup.hasOwnProperty(name))
{
var tab:BitmapButton = UIFactory.getButton(type, width, height, 0, 0, text, labelType);
tab.x = x;
tab.y = y;
tab.addEventListener(MouseEvent.CLICK, onTabClicked, false, 0, true);
_tabLookup[name] = tab;
_tabLookup[tab]
```

```
= name;
_tabs.push(tab);
addChild(tab);

//add the divider
var divider:Bitmap = UIFactory.getBitmap("TabDivider");
divider.height = tab.height;
divider.x = tab.width;
tab.addChild(divider);
layoutTabs();
}
}

public function getTab( name:String ):BitmapButton
{
if (_tabLookup.hasOwnProperty(name))
return _tabLookup[name];
return null;
}

public function setSelectedTab( name:String ):void
{
if (_tabLookup.hasOwnProperty(name))
onTabClicked(null, _tabLookup[name]);
}

public function removeTab( name:String ):void
{
if (_tabLookup.hasOwnProperty(name))
{
var tab:BitmapButton = _tabLookup[name];
tab.removeEventListener(MouseEvent.CLICK, onTabClicked);
UIFactory.destroyButton(tab);
delete _tabLookup[name];
delete _tabLookup[tab];
var index:int = _tabs.indexOf(tab);
if (index != -1)
_tabs.splice(index, 1);
removeChild(tab);
layoutTabs();
}
}

public function addSwitchTabListener( listener:Function ):void { _switchTabSignal.add(listener);
}
public function removeSwitchTabListener( listener:Function ):void {
_switchTabSignal.remove(listener); }

private function onTabClicked( e:MouseEvent = null, target:BitmapButton = null ):void
{
var
```

```
tab:BitmapButton = e ? BitmapButton(e.currentTarget) : target;
if (_tabLookup[tab])
{
for (var i:int = 0; i < _tabs.length; i++)
_tabs[i].selected = _tabs[i] == tab;
_switchTabSignal.dispatch(_tabLookup[tab]);
}
if (e)
e.stopPropagation();
}

private function layoutTabs():void
{
if (_automaticLayoutHorizontal || _automaticLayoutVertical)
{
var pos:Number = 0;
for (var i:int = 0; i < _tabs.length; i++)
{
if (_automaticLayoutHorizontal)
{
_tabs[i].x = pos;
pos += _tabs[i].width;
} else
{
_tabs[i].y = pos;
pos += _tabs[i].height;
}
}

}
}

override public function set width( value:Number ):void
{
if (_panel)
{
var bg:DisplayObject = _panel.getChildAt(0);
if (bg)
bg.width = value;
}
}

override public function set height( value:Number ):void
{
if (_panel)
{
var bg:DisplayObject = _panel.getChildAt(0);
if (bg)
bg.height = value;
}
}
```

```
public function get panelBG():DisplayObject
{
if (_panel)
{
var bg:DisplayObject = _panel.getChildAt(0);
if (bg)
return bg;
}
return null;
}

public function set automaticLayoutHorizontal( v:Boolean ):void { _automaticLayoutHorizontal =
v; _automaticLayoutVertical = false; layoutTabs(); }
public function set automaticLayoutVertical( v:Boolean ):void { _automaticLayoutVertical = v;
_automaticLayoutHorizontal = false; layoutTabs(); }

public function get enabled():Boolean { return false; }
public function set enabled( value:Boolean ):void {}

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);
_automaticLayoutHorizontal = false;
_automaticLayoutVertical = false;
_panel = UIFactory.destroyPanel(_panel);
_switchTabSignal.removeAll();
_tabLookup = null;
for (var name:String in _tabLookup)
removeTab(name);
_tabs.length = 0;
}
}
}
```

File 713: igw\com\ui\core\component\tooltips\Tooltip.as

```
package com.ui.core.component.tooltips
{
import com.ui.core.component.label.Label;

import flash.display.*;
import flash.events.*;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import
```

```
org.parade.util.DeviceMetrics;

public class Tooltip extends Sprite
{
//private static const MARGIN:int = 5;
private static const MARGIN_TOP:int = 4;
private static const MARGIN_RIGHT:int = 5;
private static const MARGIN_BOTTOM:int = 8;
private static const MARGIN_LEFT:int = 5;

private var _background:Sprite;
private var _pointer:Sprite;
private var _label:Label;
private var _layer:Sprite;
private var _width:Number;
private var _orientationUp:Boolean = true;

public function Tooltip()
{
_background = new Sprite();
_background.mouseChildren = _background.mouseEnabled = false
addChild(_background);

_pointer = new Sprite();
_pointer.graphics.lineStyle(2, 0x2c598f);
_pointer.graphics.beginFill(0x131515);
_pointer.graphics.moveTo(0, 0);
_pointer.graphics.lineTo(10, 15);
_pointer.graphics.lineTo(20, 0);
_pointer.graphics.lineTo(0, 0);
_pointer.graphics.lineStyle(2, 0x131515);
_pointer.graphics.lineTo(20, 0);
_pointer.graphics.endFill();
_pointer.mouseChildren = _pointer.mouseEnabled = false;
addChild(_pointer);

_label = new Label(12, 0xffffff, 200, 20, false, 1);
_label.constrictTextToSize = false;
_label.autoSize = TextFieldAutoSize.LEFT;
_label.align = TextFormatAlign.LEFT;
_label.mouseEnabled = false;
_background.addChild(_label);

mouseEnabled = mouseChildren = false;
}

public function init( layer:Sprite, tipText:String = "", width:Number = 200, fontSize:int = 18,
multiline:Boolean = false ):void
{
_layer = layer;
_width
```

```
= width;
//_label.fontSize = fontSize;
_label.multiline = multiline;
text = tipText;
drawBackground();
_pointer.x = -(_pointer.width / 2);
_pointer.y = -_pointer.height;
adjustToOrientation();
_layer.addChild(this);
x = _layer.mouseX;
y = (_orientationUp) ? _layer.mouseY - 4 : _layer.mouseY + 4;
addEventListener(Event.ENTER_FRAME, onEnterFrame, false, 0, true);
}

protected function drawBackground():void
{
//position the elements
var bounds:Rectangle = _background.getBounds(_background);
var offsetX:Number = 0;
var offsetY:Number = 0;
if (bounds.left > 0)
offsetX = MARGIN_RIGHT - bounds.left;
else
offsetX = bounds.left + MARGIN_LEFT;
if (bounds.top > 0)
offsetY = MARGIN_BOTTOM - bounds.top;
else
offsetY = bounds.top + MARGIN_TOP;
for (var i:int = 0; i < _background.numChildren; i++)
{
var clip:* = _background.getChildAt(i);
clip.x += offsetX;
clip.y += offsetY;
}

_background.graphics.lineStyle(2, 0x2c598f);
_background.graphics.beginFill(0x131515);
_background.graphics.drawRoundRect(0, 0, _background.width + (MARGIN_LEFT +
MARGIN_RIGHT), _background.height + (MARGIN_TOP + MARGIN_BOTTOM), 16, 16);
_background.graphics.endFill();
_background.x = -(_background.width / 2);
}

protected function adjustToOrientation():void
{
//position the pointer
_orientationUp = true;
_background.y = -(_pointer.height + _background.height) + 5;
if (_layer.mouseY - height < 0)
{
_pointer.scaleY
```

```
= -1;
_pointer.y = _pointer.height;
_background.y = _pointer.height - 5;
_orientationUp = false;
}
}

protected function onEnterFrame( e:Event = null ):void
{
_background.x = -(_background.width / 2);
var bounds:Rectangle = getBounds(_layer);
if (bounds.right > DeviceMetrics.WIDTH_PIXELS)
_background.x -= (bounds.right - DeviceMetrics.WIDTH_PIXELS);
else if (bounds.left < 0)
_background.x -= bounds.left;
x = _layer.mouseX;
y = (_orientationUp) ? _layer.mouseY - 4 : _layer.mouseY + 4;
}

public function set text( tipText:String ):void
{
_label.width = 1000;
// _label.constrictTextToSize = false;
// _label.autoSize = TextFieldAutoSize.LEFT;
//_label.text = tipText;
_label.htmlText = tipText;
_label.width = (_label.textWidth + 5 > _width) ? _width : _label.textWidth + 5;
// _label.align = TextFormatAlign.LEFT;
}

public function destroy():void
{
removeEventListener(Event.ENTER_FRAME, onEnterFrame);
_layer.removeChild(this);

_label.x = _label.y = 0;
_label.text = "";
if (!_orientationUp)
_pointer.scaleY = 1;
_orientationUp = true;
_background.graphics.clear();
}
}
}

-----------------------------------
File 714: igw\com\ui\core\component\tooltips\TooltipHandler.as
package com.ui.core.component.tooltips
{
import flash.display.InteractiveObject;
import
```

```
flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.utils.Timer;

import org.as3commons.logging.api.ILogger;
import org.as3commons.logging.api.getLogger;
import org.shared.ObjectPool;

public class TooltipHandler
{
private const logger:ILogger = getLogger('display.core.component.tooltips');

private var _callback:Function;
private var _layer:Sprite;
private var _showTimer:Timer;
private var _target:InteractiveObject;
private var _text:String;
private var _tip:Tooltip;
private var _width:Number;
private var _fontSize:int;
private var _multiline:Boolean;

public function init( layer:Sprite, target:InteractiveObject, callback:Function = null, text:String =
null, delay:int = 500, width:Number = 200, fontSize:int = 18, multiline:Boolean = false ):void
{
_layer = layer;
_target = target;
_target.addEventListener(MouseEvent.MOUSE_OVER, onMouseOver, false, 0, true);
_target.addEventListener(MouseEvent.MOUSE_OUT, onMouseOut, false, 0, true);
_target.addEventListener(MouseEvent.MOUSE_DOWN, onMouseOut, false, 0, true);
if (!_showTimer)
{
_showTimer = new Timer(500, 1);
_showTimer.addEventListener(TimerEvent.TIMER_COMPLETE, showTooltip);
}
_showTimer.delay = delay;
_callback = callback;
_text = text;
_width = width;
_fontSize = fontSize;
_multiline = multiline;
}

private function onMouseOver( e:MouseEvent ):void
{
_showTimer.start();
}

private function onMouseOut( e:MouseEvent = null ):void
{
```

```
if (_showTimer)
_showTimer.reset();

if (_tip)
{
ObjectPool.give(_tip);
_tip = null;
}
}

private function showTooltip( e:TimerEvent ):void
{
/*try
{*/
var str:String;
if (_callback != null)
{
// str = _callback();
str = _callback.apply(_target);
}

else if (_text)
str = _text;

if (str && str != "")
{
_tip = ObjectPool.get(Tooltip);
_tip.init(_layer, str, _width, _fontSize, _multiline);
}
/*} catch ( e:Error )
{
logger.error("Tooltip failed to initialize.")
}*/

}

public function get target():* { return _target; }

public function destroy():void
{
onMouseOut();
if (_target)
{
_target.removeEventListener(MouseEvent.MOUSE_OVER, onMouseOver);
_target.removeEventListener(MouseEvent.MOUSE_OUT, onMouseOut);
_target.removeEventListener(MouseEvent.MOUSE_DOWN, onMouseOut);
_showTimer.removeEventListener(TimerEvent.TIMER_COMPLETE, showTooltip);
_layer = null;
_target = null;
_callback
```

```
= null;
_showTimer = null;
}
}
}
}

-----------------------------------
File 715: igw\com\ui\core\component\tooltips\Tooltips.as
package com.ui.core.component.tooltips
{
import flash.utils.Dictionary;

import org.parade.core.IViewStack;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class Tooltips
{
private static const TOOLTIP_SHOW_DELAY:int = 250;
private static const DEFAULT_WIDTH:int = 180;
private static const DEFAULT_FONT_SIZE:int = 18;

private var _tooltips:Dictionary = new Dictionary(true);

[Inject]
public var viewStack:IViewStack;

public function addTooltip( target:*,
parent:* = null,
callback:Function = null,
text:String = "",
delay:int = TOOLTIP_SHOW_DELAY,
width:Number = DEFAULT_WIDTH,
fontSize:int = DEFAULT_FONT_SIZE,
multiline:Boolean = false ):void
{
if (_tooltips.hasOwnProperty(target))
return;

var handler:TooltipHandler = ObjectPool.get(TooltipHandler);
handler.init(viewStack.getLayer(ViewEnum.HOVER), target, callback, text, delay, width,
fontSize, multiline);
_tooltips[target] = handler;
if (parent)
{
if (_tooltips[parent] == null)
_tooltips[parent] = [];
if (_tooltips[parent] is Array)
_tooltips[parent].push(handler);
}
```

```
}

public function removeTooltip( target:*, parent:* = null ):void
{
if (parent && _tooltips[parent] != null && _tooltips[parent] is Array)
{
var handlers:Array = _tooltips[parent];
var handler:TooltipHandler;
for (var i:int = 0; i < handlers.length; i++)
{
handler = handlers[i];
if (_tooltips.hasOwnProperty(handler.target))
delete _tooltips[handler.target];
ObjectPool.give(handler);
}
_tooltips[parent] = null;
delete _tooltips[parent];

} else if (_tooltips[target])
{
ObjectPool.give(_tooltips[target]);
_tooltips[target] = null;
delete _tooltips[target];
}
}
}
}

-----------------------------------
File 716: igw\com\ui\core\component\videoplayer\ControlPanel.as
package com.ui.core.component.videoplayer
{
import com.Application;
import com.ui.core.component.button.BitmapButton;
import com.ui.modal.ButtonFactory;

import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.ui.Mouse;
import flash.utils.getDefinitionByName;

public class ControlPanel extends Sprite
{
private var _controlPanelBG:Sprite;

private var _playBtn:BitmapButton;
private var _pauseBtn:BitmapButton;
private
```

```
var _fullScreenBtn:BitmapButton;

private var _growUp:BitmapData;
private var _growRollOver:BitmapData;

private var _shrinkUp:BitmapData;
private var _shrinkRollOver:BitmapData;

private var _isPlaying:Boolean;
private var _isFullScreen:Boolean;

private var _fullScreenFunction:Function;
public var startFunction:Function;
public var pauseFunction:Function;
public var resumeFunction:Function;

public function ControlPanel( fullScreenFunction:Function = null)
{
super();
_controlPanelBG = new Sprite();

_playBtn = ButtonFactory.getBitmapButton('VideoPlayerPlayBtnUpBMD', 0, 0, '', 0,
'VideoPlayerPlayBtnRollOverBMD', 'VideoPlayerPlayBtnDownBMD');
_playBtn.addEventListener(MouseEvent.CLICK, onPlayBtnClicked, false, 0, true);
_playBtn.visible = false;

_pauseBtn = ButtonFactory.getBitmapButton('VideoPlayerPauseBtnUpBMD', 0, 0, '', 0,
'VideoPlayerPauseBtnRollOverBMD');
_pauseBtn.addEventListener(MouseEvent.CLICK, onPauseBtnClicked, false, 0, true);
_pauseBtn.visible = false;

addChild(_controlPanelBG);
addChild(_playBtn);
addChild(_pauseBtn);
}

public function init( width:Number, height:Number ):void
{
_controlPanelBG.graphics.clear();
_controlPanelBG.graphics.lineStyle(2, 0x2c598f);
_controlPanelBG.graphics.beginFill(0x131515);
_controlPanelBG.graphics.moveTo(1, 0);
_controlPanelBG.graphics.lineTo(width, 0);
_controlPanelBG.graphics.lineTo(width, height * 0.1);
_controlPanelBG.graphics.lineTo(1, height * 0.1);
_controlPanelBG.graphics.lineStyle(2, 0x131515);
_controlPanelBG.graphics.lineTo(1, 0);
_controlPanelBG.graphics.endFill();
_controlPanelBG.alpha = 0.4;
_controlPanelBG.cacheAsBitmap = true;

_playBtn.scaleX
```

```
= 1;
_playBtn.scaleY = 1;

while(_playBtn.height > _controlPanelBG.height)
{
_playBtn.scaleX -= 0.1;
_playBtn.scaleY -= 0.1;
}

_pauseBtn.scaleX = 1;
_pauseBtn.scaleY = 1;

while(_pauseBtn.height > _controlPanelBG.height)
{
_pauseBtn.scaleX -= 0.1;
_pauseBtn.scaleY -= 0.1;
}

_playBtn.x = _controlPanelBG.x + (_controlPanelBG.width - _playBtn.width) * 0.5;
_playBtn.y = _controlPanelBG.y + (_controlPanelBG.height - _playBtn.height) * 0.5;

_pauseBtn.height = height * 0.1 - 5;
_pauseBtn.x = _controlPanelBG.x + (_controlPanelBG.width - _pauseBtn.width) * 0.5;
_pauseBtn.y = _controlPanelBG.y + (_controlPanelBG.height - _pauseBtn.height) * 0.5;

if(_fullScreenFunction != null)
{
_fullScreenBtn.scaleX = 1;
_fullScreenBtn.scaleY = 1;

while(_fullScreenBtn.height > _controlPanelBG.height)
{
_fullScreenBtn.scaleX -= 0.1;
_fullScreenBtn.scaleY -= 0.1;
}

_fullScreenBtn.y = _pauseBtn.y + (_pauseBtn.height - _fullScreenBtn.height) * 0.5
_fullScreenBtn.x = _controlPanelBG.width - _fullScreenBtn.width - 20;
}

}

override public function get height():Number
{
return _controlPanelBG.height;
}

override public function get width():Number
{
return _controlPanelBG.width;
}
```

```
public function set isPlaying( isPlaying:Boolean ):void
{
_isPlaying = isPlaying;
if(_isPlaying)
{
_pauseBtn.visible = true;
_playBtn.visible = false;
}
else
{
_pauseBtn.visible = false;
_playBtn.visible = true;
}
}

private function onPlayBtnClicked( e:MouseEvent ):void
{
play();
}

public function play():void
{
if(!_isPlaying)
{
startFunction();
_isPlaying = true;
}

if(resumeFunction != null)
resumeFunction();

_playBtn.visible = false;
_pauseBtn.visible = true;
}

private function onPauseBtnClicked( e:MouseEvent ):void
{
if(pauseFunction != null)
pauseFunction();

_pauseBtn.visible = false;
_playBtn.visible = true;
}

private function onFullScreen( e:MouseEvent ):void
{
_fullScreenFunction(!_isFullScreen);
}
```

```
public function set fullScreen( isFullScreen:Boolean ):void
{
_isFullScreen = isFullScreen;

if(_isFullScreen)
_fullScreenBtn.updateBackgrounds(_shrinkUp, _shrinkRollOver);
else
_fullScreenBtn.updateBackgrounds(_growUp, _growRollOver);
}

public function destroy():void
{
_controlPanelBG = null;

_playBtn.removeEventListener(MouseEvent.CLICK, onPlayBtnClicked);
_playBtn.destroy();
_playBtn = null;

_pauseBtn.removeEventListener(MouseEvent.CLICK, onPauseBtnClicked);
_pauseBtn.destroy();
_pauseBtn = null;

if(_fullScreenFunction != null)
{
_fullScreenBtn.removeEventListener(MouseEvent.CLICK, onFullScreen);
_fullScreenBtn.destroy();
_fullScreenBtn = null;
}
}

public function set fullScreenFunction( fullScreenFunction:Function ):void
{
_fullScreenFunction = fullScreenFunction;
_fullScreenBtn = ButtonFactory.getBitmapButton('IconMaximizeBMD', 369, 5, '', 0,
'IconMaximizeRollOverBMD');
_fullScreenBtn.addEventListener(MouseEvent.CLICK, onFullScreen, false, 0, true);

var growUpBG:Class = Class(getDefinitionByName('IconMaximizeBMD'));
var growRollOverBG:Class = Class(getDefinitionByName('IconMaximizeRollOverBMD'));

var shrinkBG:Class = Class(getDefinitionByName('IconMinimizeBMD'));
var shrinkRollOverBG:Class = Class(getDefinitionByName('IconMinimizeRollOverBMD'));

_growUp = BitmapData(new growUpBG());
_growRollOver= BitmapData(new growRollOverBG());

_shrinkUp = BitmapData(new shrinkBG());
_shrinkRollOver = BitmapData(new shrinkRollOverBG());


addChild(_fullScreenBtn);
```

```
}
}
}
```

File 717: igw\com\ui\core\component\videoplayer\VideoPlayer.as
```
package com.ui.core.component.videoplayer
{
import com.Application;
import com.ui.core.component.button.BitmapButton;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.display.StageDisplayState;
import flash.events.AsyncErrorEvent;
import flash.events.Event;
import flash.events.FullScreenEvent;
import flash.events.MouseEvent;
import flash.events.NetStatusEvent;
import flash.events.SecurityErrorEvent;
import flash.events.StageVideoAvailabilityEvent;
import flash.events.TimerEvent;
import flash.geom.Rectangle;
import flash.media.SoundTransform;
import flash.media.StageVideo;
import flash.media.Video;
import flash.net.NetConnection;
import flash.net.NetStream;
import flash.utils.Timer;
import flash.utils.getDefinitionByName;

import org.greensock.TweenLite;
import org.starling.core.Starling;

public class VideoPlayer extends Sprite
{
private var _videoHolder:Sprite;
private var _bufferingImageContainer:Sprite;

private var _fullPlayBtn:BitmapButton;

private var _controlPanel:ControlPanel;

private var _video:Video;
private var _stageVideo:StageVideo;
private var _videoConnection:NetConnection;

private var _netStream:NetStream;

private
```

```
var _videoURL:String;
private var _server:String;

private var _defaultWidth:Number;
private var _defaultHeight:Number;

private var _width:Number;
private var _height:Number;
private var _duration:Number;

private var _onMouseInactivity:Timer;

private var _init:Boolean;
private var _startVideoImmediately:Boolean;
private var _loop:Boolean;
private var _fullScreen:Boolean;

private var _nonFullScreenX:int;
private var _nonFullScreenY:int;

private var _bufferingImage:Bitmap;

private var _stageVideoAvailability:String;

private var _frameToPauseOn:Number;

public var onVideoEnd:Function;
public var onVideoFullScreen:Function;

private var _volume:Number;

public function VideoPlayer( server:String, videoURL:String, vidWidth:Number = 0,
vidHeight:Number = 0, startVideoImmediately:Boolean = false, loop:Boolean = false,
frameToPauseOn:Number = 0, volume:Number =
1 )
{
_volume = volume;
_loop = loop;
_videoURL = videoURL;
_server = server;
_defaultWidth = _width = vidWidth;
_defaultHeight = _height = vidHeight;
_startVideoImmediately = startVideoImmediately;
_frameToPauseOn = frameToPauseOn;

_fullPlayBtn = ButtonFactory.getBitmapButton('VideoPlayerPlayBtnUpBMD', 0, 0, '', 0,
'VideoPlayerPlayBtnRollOverBMD', 'VideoPlayerPlayBtnDownBMD');
_fullPlayBtn.addEventListener(MouseEvent.CLICK, onPlayBtnClicked, false, 0, true);
_fullPlayBtn.visible = false;

_controlPanel
```

```
= new ControlPanel();
_controlPanel.alpha = 0;
_controlPanel.startFunction = start;
_controlPanel.pauseFunction = pause;
_controlPanel.resumeFunction = resume;
if (Application.STAGE.hasOwnProperty('displayState'))
{
_controlPanel.fullScreenFunction = onFullScreen;
Application.STAGE.addEventListener(Event.FULLSCREEN, onFullScreenSet, false, 0, true);
}


Application.STAGE.addEventListener(StageVideoAvailabilityEvent.STAGE_VIDEO_AVAILABILITY,
onStageVideoAvailability);

_controlPanel.addEventListener(MouseEvent.MOUSE_MOVE, onMouseMove, false, 0, true);

_onMouseInactivity = new Timer(2000, 1);
_onMouseInactivity.addEventListener(TimerEvent.TIMER_COMPLETE, onMouseInactive, false,
0, true);

_videoHolder = new Sprite();
_videoHolder.addEventListener(MouseEvent.MOUSE_MOVE, onMouseMove, false, 0, true);

var bufferingBMD:Class = Class(getDefinitionByName(('VideoPlayerBufferIconBMD')));
_bufferingImage = new Bitmap(BitmapData(new bufferingBMD()));
_bufferingImage.x = -_bufferingImage.width * 0.5;
_bufferingImage.y = -_bufferingImage.height * 0.5;
_bufferingImage.smoothing = true;

_bufferingImageContainer = new Sprite();
_bufferingImageContainer.x = vidWidth * 0.5;
_bufferingImageContainer.y = vidHeight * 0.5;
_bufferingImageContainer.addChild(_bufferingImage);
_bufferingImageContainer.visible = false;

_fullPlayBtn.x = (vidWidth - _fullPlayBtn.width) * 0.5;
_fullPlayBtn.y = (vidHeight - _fullPlayBtn.height) * 0.5;

_videoConnection = new NetConnection();
_videoConnection.addEventListener(NetStatusEvent.NET_STATUS, onNetStatus, false, 0,
true);
_videoConnection.addEventListener(SecurityErrorEvent.SECURITY_ERROR,
securityErrorHandler, false, 0, true);
if (server != '')
_videoConnection.connect(server);
else
_videoConnection.connect(null);

onBuffering(true);
}
```

```
private function onPlayBtnClicked( e:MouseEvent ):void
{
_controlPanel.play();
}

private function onNetStatus( event:Object ):void
{
switch (event.info.code)
{
case "NetConnection.Connect.Success":
connectStream();
break;
case "NetStream.Buffer.Empty":
onBuffering(true);
break;
case "NetStream.Buffer.Full":
onBuffering(false);
break;
case "NetStream.Play.StreamNotFound":
trace("Stream not found: " + _videoURL);
break;
case "NetStream.Unpause.Notify":
_fullPlayBtn.visible = false;
break;
case "NetStream.Play.Start":
_fullPlayBtn.visible = false;
if (!_startVideoImmediately)
_netStream.pause();
break;
case "NetStream.Pause.Notify":
_fullPlayBtn.visible = true;
break;
case "NetStream.Play.Stop":
_fullPlayBtn.visible = true;
videoPlayComplete();
break;
}
}

private function onBuffering( buffering:Boolean ):void
{
if (buffering)
{
TweenLite.to(_bufferingImageContainer, 300, {rotation:'54000'});
} else
{
TweenLite.killTweensOf(_bufferingImageContainer);
}

_bufferingImageContainer.visible
```

```
= buffering;
}

private function connectStream():void
{
var soundTransform:SoundTransform = new SoundTransform();
soundTransform.volume = _volume;

_netStream = new NetStream(_videoConnection);
_netStream.addEventListener(AsyncErrorEvent.ASYNC_ERROR, asyncErrorHandler, false, 0,
true);
_netStream.addEventListener(NetStatusEvent.NET_STATUS, onNetStatus, false, 0, true);
_netStream.bufferTime = Application.AVG_LOAD_TIME;
_netStream.soundTransform = soundTransform;


var custom_obj:Object = new Object();
custom_obj.onMetaData = onMetaDataHandler;
custom_obj.onCuePoint = onCuePointHandler;
custom_obj.onPlayStatus = playStatus;
_netStream.client = custom_obj;

_video = new Video();

addChild(_videoHolder);
addChild(_bufferingImageContainer);
addChild(_fullPlayBtn);
addChild(_controlPanel);
setVideoInit();
}

private function disableStageVideo():void
{
_video.attachNetStream(_netStream);
addChildAt(_video, 0);
}

private function enableStageVideo():void
{

if (_stageVideo == null)
{
_stageVideo = Application.STAGE.stageVideos[0];
_stageVideo.viewPort = new Rectangle(x, y, _width, _height);

if (Application.STARLING_ENABLED)
Starling.current.stage3D.visible = false;
}

if (_video && _video.parent)
removeChild(_video);
```

```
_stageVideo.attachNetStream(_netStream);
}

override public function set x( value:Number ):void
{
if (_stageVideo != null)
_stageVideo.viewPort = new Rectangle(value, y, _width, _height);

super.x = value;
}

override public function set y( value:Number ):void
{
if (_stageVideo != null)
_stageVideo.viewPort = new Rectangle(x, value, _width, _height);
super.y = value;
}


private function onStageVideoAvailability( e:StageVideoAvailabilityEvent ):void
{
_stageVideoAvailability = e.availability;
if (_stageVideoAvailability && Application.STAGE.stageVideos.length > 0)
enableStageVideo();
else
disableStageVideo();
}


public function updateVideo( videoURL:String, frameToPauseOn:Number = 0, volume:Number =
1 ):void
{
if (_videoURL != videoURL)
{
_videoURL = videoURL;
_frameToPauseOn = frameToPauseOn;
_volume = volume;
TweenLite.killTweensOf(_controlPanel);
_controlPanel.alpha = 0;
cleanUpOldVideo();
connectStream();

if (_stageVideoAvailability && Application.STAGE.stageVideos.length > 0)
enableStageVideo();
else
disableStageVideo();
}
}

private
```

```
function cleanUpOldVideo():void
{
_init = false;
if (_video.parent)
removeChild(_video);

_stageVideo = null;
_video.clear();
_video = null;

_netStream.removeEventListener(AsyncErrorEvent.ASYNC_ERROR, asyncErrorHandler);
_netStream.removeEventListener(NetStatusEvent.NET_STATUS, onNetStatus);
_netStream.close();
_netStream = null;
}

private function onMouseInactive( e:TimerEvent ):void
{
TweenLite.to(_controlPanel, 1, {y:(_video.height), alpha:0});
}

private function onMouseMove( e:MouseEvent ):void
{
TweenLite.to(_controlPanel, 1, {y:(_video.height - _controlPanel.height), alpha:1});

_onMouseInactivity.reset();
_onMouseInactivity.start();
}

private function onMetaDataHandler( metadata:Object ):void
{
if (!_init)
{
_init = true;
if (_width == 0)
{
_defaultWidth = metadata.width;
}

if (_height == 0)
{
_defaultHeight = metadata.height;
}

resize(_defaultWidth, _defaultHeight);
_duration = metadata.duration;

_controlPanel.isPlaying = _startVideoImmediately;

if (!_startVideoImmediately)
_netStream.seek(_frameToPauseOn);
```

```
}

}

public function resize( width:Number, height:Number ):void
{
if (_stageVideo != null)
_stageVideo.viewPort = new Rectangle(x, y, width, height);

_videoHolder.graphics.clear();
_videoHolder.graphics.lineStyle(2, 0xffffff, 0);
_videoHolder.graphics.beginFill(0xffffff, 0);
_videoHolder.graphics.moveTo(0, 0);
_videoHolder.graphics.lineTo(width, 0);
_videoHolder.graphics.lineTo(width, height);
_videoHolder.graphics.lineTo(0, height);
_videoHolder.graphics.lineTo(0, 0);
_videoHolder.graphics.endFill();

_video.width = width;
_video.height = height;

setUpControlPanel(width, height);
}

private function setUpControlPanel( width:Number, height:Number ):void
{
_controlPanel.init(width, height);

TweenLite.killTweensOf(_controlPanel);
if (_onMouseInactivity.running)
_controlPanel.y = height - _controlPanel.height;
else
_controlPanel.y = height;
}


override public function get height():Number
{
return _height;
}

override public function get width():Number
{
return _width;
}


private function securityErrorHandler( event:SecurityErrorEvent ):void
{
```

```
trace("securityErrorHandler: " + event);
}
private function asyncErrorHandler( event:AsyncErrorEvent ):void
{
trace(event.text);
}

private function onCuePointHandler( cueInfoObj:Object ):void
{

}

private function videoPlayComplete():void
{
_init = false;

_netStream.seek(_frameToPauseOn);

if (!_loop)
_netStream.pause();

if (_controlPanel)
_controlPanel.isPlaying = _loop;

if (onVideoEnd != null)
onVideoEnd();
}

private function setVideoInit():void
{
_netStream.play(_videoURL);
}

private function playStatus( event:Object ):void
{
switch (event.code)
{
case "NetStream.Play.Complete":
videoPlayComplete();
break;
}
}

private function onPlay( e:MouseEvent ):void
{
play();
}

private function onPause( e:MouseEvent ):void
{
```

```
pause();
}

private function onResume( e:MouseEvent ):void
{
resume();
}

private function start():void
{
_netStream.seek(0);
}

public function play():void
{
_netStream.resume();
}

public function pause():void
{
_netStream.pause();
}

public function resume():void
{
_netStream.resume();
}

private function onFullScreen( fullScreen:Boolean ):void
{
if (fullScreen)
Application.STAGE.displayState = StageDisplayState.FULL_SCREEN_INTERACTIVE;
else
Application.STAGE.displayState = StageDisplayState.NORMAL;
}

private function onFullScreenSet( e:FullScreenEvent ):void
{
var xPos:Number;
var yPos:Number;
_fullScreen = e.fullScreen;
if (e.fullScreen)
{
_nonFullScreenX = x;
_nonFullScreenY = y;

xPos = 0;
yPos = 0;

_width
```

```
= Application.STAGE.fullScreenWidth;
_height = Application.STAGE.fullScreenHeight;

} else
{
xPos = _nonFullScreenX;
yPos = _nonFullScreenY;

_width = _defaultWidth;
_height = _defaultHeight;

}

x = xPos;
y = yPos;

_controlPanel.fullScreen = e.fullScreen;
resize(_width, _height);

_fullPlayBtn.x = (_width - _fullPlayBtn.width) * 0.5;
_fullPlayBtn.y = (_height - _fullPlayBtn.height) * 0.5;

if (onVideoFullScreen != null)
onVideoFullScreen(_fullScreen);
}

public function destroy():void
{

Application.STAGE.removeEventListener(StageVideoAvailabilityEvent.STAGE_VIDEO_AVAILABILITY,
onStageVideoAvailability);

if (Application.STAGE.hasOwnProperty('displayState'))
Application.STAGE.removeEventListener(Event.FULLSCREEN, onFullScreenSet);

if (_fullScreen)
{
Application.STAGE.displayState = StageDisplayState.NORMAL;
}

TweenLite.killTweensOf(_controlPanel);
//_controlPanel.alpha = 0;
_controlPanel.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
_controlPanel.destroy();
_controlPanel = null;
if (_onMouseInactivity.running)
_onMouseInactivity.stop();

_onMouseInactivity.removeEventListener(TimerEvent.TIMER_COMPLETE, onMouseInactive);

_videoHolder.removeEventListener(MouseEvent.MOUSE_MOVE,
```

```
onMouseMove);

_stageVideo = null;

_video.clear();
_video = null;

_netStream.removeEventListener(AsyncErrorEvent.ASYNC_ERROR, asyncErrorHandler);
_netStream.removeEventListener(NetStatusEvent.NET_STATUS, onNetStatus);
_netStream.close();
_netStream.dispose();
_netStream = null;

_videoConnection.removeEventListener(NetStatusEvent.NET_STATUS, onNetStatus);
_videoConnection.removeEventListener(SecurityErrorEvent.SECURITY_ERROR,
securityErrorHandler);
_videoConnection.close();
_videoConnection = null;

_fullPlayBtn.removeEventListener(MouseEvent.CLICK, onPlayBtnClicked);
_fullPlayBtn.destroy();
_fullPlayBtn = null;

_videoHolder = null;

_bufferingImage = null;

_bufferingImageContainer = null;

if (Application.STARLING_ENABLED)
Starling.current.stage3D.visible = true;
}
}
}

-----------------------------------
File 718: igw\com\ui\core\component\videoplayer\YouTubeVideoPlayer.as
ï»¿package com.ui.core.component.videoplayer
{
import com.Application;
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.modal.ButtonFactory;

import flash.display.Loader;
import flash.display.Sprite;
import flash.display.StageDisplayState;
import flash.events.Event;
import flash.events.FullScreenEvent;
import
```

```
flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.net.URLRequest;
import flash.system.ApplicationDomain;
import flash.system.LoaderContext;
import flash.system.Security;
import flash.utils.Timer;

import org.greensock.TweenLite;
import org.osflash.signals.Signal;

public class YouTubeVideoPlayer extends Sprite
{
private var _hitArea:Sprite;
private var _playBtn:BitmapButton;
private var _pauseBtn:BitmapButton;
private var _minimizeBtn:BitmapButton;
private var _fullScreenBtn:BitmapButton;

private var _vPlayer:Object;
private var _vLoader:Loader;
private var _vName:String;

private var _vVolume:uint;

private var _vCurrentState:int;

private var _defaultWidth:Number;
private var _defaultHeight:Number;

private var _width:Number;
private var _height:Number;

private var _autoplay:Boolean;
private var _fullScreen:Boolean;
private var _isPlaying:Boolean;
private var _isReady:Boolean;

private var _onMouseInactivity:Timer;
private var _onVideoEnd:Function;

public var onFullScreenChanged:Signal;

private var UNSTARTED:int = -1;
private var ENDED:int = 0;
private var PLAYING:int = 1;
private var PAUSED:int = 2;
private var BUFFERING:int = 3;
private var VIDEO_CUED:int = 4;

public
```

```
function YouTubeVideoPlayer( width:Number, height:Number, autoplay:Boolean = false )
{

if(CONFIG::IS_DESKTOP == 0)
Security.allowDomain("www.youtube.com");



onFullScreenChanged = new Signal(Boolean);

_hitArea = new Sprite();
_hitArea.alpha = 0;

_autoplay = autoplay;

_defaultWidth = _width = width;
_defaultHeight = _height = height;

drawHitArea();

_vCurrentState = -1;

this.addEventListener(MouseEvent.MOUSE_MOVE, onMouseMove, false, 0, true);

//_vLoader = new Loader();
//_vLoader.load(new URLRequest("https://www.youtube.com/apiplayer?version=3"), new
LoaderContext(false, ApplicationDomain.currentDomain));
//_vLoader.contentLoaderInfo.addEventListener(Event.INIT, onLoaderInit, false, 0, true);

_onMouseInactivity = new Timer(2000, 1);
_onMouseInactivity.addEventListener(TimerEvent.TIMER_COMPLETE, onMouseInactive, false,
0, true);

_playBtn = ButtonFactory.getBitmapButton('VideoPlayerPlayBtnUpBMD', 0, 0, '', 0,
'VideoPlayerPlayBtnRollOverBMD', 'VideoPlayerPlayBtnDownBMD');
_playBtn.addEventListener(MouseEvent.CLICK, onPlayVideoClick, false, 0, true);
_playBtn.visible = !autoplay;

_pauseBtn = ButtonFactory.getBitmapButton('VideoPlayerPauseBtnUpBMD', 0, 0, '', 0,
'VideoPlayerPauseBtnRollOverBMD', 'VideoPlayerPauseBtnDownBMD');
_pauseBtn.addEventListener(MouseEvent.CLICK, onPauseVideoClicked, false, 0, true);
_playBtn.visible = autoplay;
_pauseBtn.alpha = 0;

_fullScreenBtn = UIFactory.getButton(ButtonEnum.ICON_MAXIMIZE, 0, 0, 369, 5);
_fullScreenBtn.addEventListener(MouseEvent.CLICK, onFullScreenClick, false, 0, true);
_fullScreenBtn.alpha = 0;
_fullScreenBtn.visible = true;

_minimizeBtn = UIFactory.getButton(ButtonEnum.ICON_MINIMIZE, 0, 0, 369, 5);
_minimizeBtn.addEventListener(MouseEvent.CLICK,
```

```
onFullScreenClick, false, 0, true);
_minimizeBtn.alpha = 0;
_minimizeBtn.visible = false;

Application.STAGE.addEventListener(Event.FULLSCREEN, onFullScreenSet, false, 0, true);
addChild(_hitArea);
}

private function onLoaderInit( e:Event ):void
{
_vPlayer = _vLoader.content;
_vPlayer.addEventListener('onReady', onPlayerReady, false, 0, true);
_vPlayer.addEventListener('onStateChange', onStateChanged, false, 0, true);

addChild(_vLoader);
addChild(_hitArea);
addChild(_fullScreenBtn);
addChild(_minimizeBtn);
addChild(_pauseBtn);
addChild(_playBtn);
}

private function drawHitArea():void
{
if (_hitArea)
{
_hitArea.graphics.clear();
_hitArea.graphics.beginFill(0xffffff, 0.0);
_hitArea.graphics.drawRect(0, 0, _width, _height);
_hitArea.graphics.endFill();
}
}

private function onPlayerReady( e:Event ):void
{
_isReady = true;
_vPlayer.setVolume(_vVolume);
resize();
loadVideo();
}

public function resizeDefault( width:Number, height:Number ):void
{
_defaultWidth = _width = width;
_defaultHeight = _height = height;
resize();
}

private function resize():void
{
drawHitArea();
```

```
_vPlayer.setSize(_width, _height);

_playBtn.x = (_width - _playBtn.width) * 0.5;
_playBtn.y = (_height - _playBtn.height) * 0.5;

_pauseBtn.x = (_width - _pauseBtn.width) * 0.5;
_pauseBtn.y = (_height - _pauseBtn.height) * 0.5;

_fullScreenBtn.x = _minimizeBtn.x = _fullScreenBtn.width;
_fullScreenBtn.y = _minimizeBtn.y = _height - _fullScreenBtn.height - 10;
}

private function loadVideo():void
{
if (_vName != '' && _isReady)
{
if (!_autoplay)
_vPlayer.cueVideoById(_vName, 0);
else
_vPlayer.loadVideoById(_vName, 0);
}
}

private function onFullScreen( fullScreen:Boolean ):void
{
if (fullScreen)
Application.STAGE.displayState = StageDisplayState.FULL_SCREEN_INTERACTIVE;
else
Application.STAGE.displayState = StageDisplayState.NORMAL;
}

private function onFullScreenSet( e:FullScreenEvent ):void
{
_fullScreen = e.fullScreen;

if (e.fullScreen)
{
_width = Application.STAGE.fullScreenWidth;
_height = Application.STAGE.fullScreenHeight;
_fullScreenBtn.visible = false;
_minimizeBtn.visible = true;
} else
{
_width = _defaultWidth;
_height = _defaultHeight;
_fullScreenBtn.visible = true;
_minimizeBtn.visible = false;
}

onFullScreenChanged.dispatch(_fullScreen);
```

```
resize();
}

private function onMouseMove( e:MouseEvent ):void
{
TweenLite.to(_fullScreenBtn, 1, {alpha:1});
TweenLite.to(_minimizeBtn, 1, {alpha:1});

if (_isPlaying)
TweenLite.to(_pauseBtn, 1, {alpha:1});

_onMouseInactivity.reset();
_onMouseInactivity.start();
}

private function onMouseInactive( e:TimerEvent ):void
{
TweenLite.to(_fullScreenBtn, 1, {alpha:0});
TweenLite.to(_minimizeBtn, 1, {alpha:0});

if (_pauseBtn.alpha > 0)
TweenLite.to(_pauseBtn, 1, {alpha:0});
}

private function onStateChanged( e:Event ):void
{
_vCurrentState = _vPlayer.getPlayerState();
switch (_vCurrentState)
{
case UNSTARTED:
_isPlaying = false;
_playBtn.visible = !_isPlaying;
break;
case ENDED:
_isPlaying = false;
if (_onVideoEnd != null)
_onVideoEnd();
break;
case PLAYING:
_isPlaying = true;
_pauseBtn.visible = true;
_playBtn.visible = !_isPlaying;
break;
case PAUSED:
_isPlaying = false;
_pauseBtn.visible = _isPlaying;
_playBtn.visible = !_isPlaying;
break;
case BUFFERING:
_isPlaying
```

```
= false;
_playBtn.visible = !_isPlaying;
break;
case VIDEO_CUED:
_isPlaying = false;
_playBtn.visible = !_isPlaying;
break;
}
}

public function updateVideo( v:String ):void
{
if (_vPlayer)
_vPlayer.stopVideo();

_vName = v;

loadVideo();
}

public function stopVideo():void
{
_vPlayer.stopVideo();
}

public function onFullScreenClick( e:MouseEvent ):void
{
onFullScreen(!_fullScreen)
}

public function onPlayVideoClick( e:MouseEvent ):void
{
_vPlayer.playVideo();
}

private function onPauseVideoClicked( e:MouseEvent ):void
{
_vPlayer.pauseVideo();
}

public function set onVideoEnd( v:Function ):void { _onVideoEnd = v; }

public function set volume( v:uint ):void
{
_vVolume = v;

if (_isReady)
_vPlayer.setVolume(_vVolume);
}

public
```

```
function destroy():void
{
this.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
Application.STAGE.removeEventListener(Event.FULLSCREEN, onFullScreenSet);
if (_vCurrentState == PLAYING || _vCurrentState == BUFFERING)
_vPlayer.stopVideo();

if (onFullScreenChanged)
onFullScreenChanged.removeAll();
onFullScreenChanged = null;

if (_playBtn)
{
_playBtn.removeEventListener(MouseEvent.CLICK, onPlayVideoClick);
_playBtn.destroy();
}

_playBtn = null;

if (_pauseBtn)
{
_pauseBtn.removeEventListener(MouseEvent.CLICK, onPauseVideoClicked);
_pauseBtn.destroy();
}

_pauseBtn = null;

_fullScreenBtn.removeEventListener(MouseEvent.CLICK, onFullScreenClick);
_minimizeBtn.removeEventListener(MouseEvent.CLICK, onFullScreenClick);
_fullScreenBtn = UIFactory.destroyButton(_fullScreenBtn);
_minimizeBtn = UIFactory.destroyButton(_minimizeBtn);

if (_vPlayer)
{
_vPlayer.removeEventListener('onReady', onPlayerReady);
_vPlayer.removeEventListener('onStateChange', onStateChanged);
}

_vPlayer = null;

if (_vLoader)
_vLoader.contentLoaderInfo.removeEventListener(Event.INIT, onLoaderInit);

_vLoader = null;

_vName = null;

if (_onMouseInactivity)
{
if (_onMouseInactivity.running)
_onMouseInactivity.stop();
```

```
_onMouseInactivity.removeEventListener(TimerEvent.TIMER_COMPLETE, onMouseInactive);
}
_onMouseInactivity = null;

_onVideoEnd = null;

_hitArea = null;
}

}
}

-----------------------------------
File 719: igw\com\ui\core\effects\AlphaEffect.as
ï»¿package com.ui.core.effects
{
import org.greensock.TweenLite;
import org.parade.core.IView;

public class AlphaEffect extends Effect
{
public static const NAME:String = "alphaEffect";

private var _start:Number;
private var _middle:Number;
private var _end:Number;
private var _timeIN:Number;
private var _timeOUT:Number;

public function init( start:Number, middle:Number, end:Number, timeIN:Number,
timeOUT:Number ):void
{
_name = NAME;
_start = start;
_middle = middle;
_end = end;
_timeIN = timeIN;
_timeOUT = timeOUT;
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
screen.alpha = _start;
TweenLite.to(screen, _timeIN, {alpha:_middle, onComplete:doneIn, overwrite:false});
}

override internal function goOut( screen:IView ):void
{
```

```
super.goOut(screen);
TweenLite.to(screen, _timeOUT, {alpha:_end, onComplete:doneOut, overwrite:false});
}
}
}
```

File 720: igw\com\ui\core\effects\Effect.as

```
package com.ui.core.effects
{
import org.parade.core.IView;

public class Effect
{
protected var _inCallback:Function;
protected var _outCallback:Function;
protected var _name:String;

internal function goIn( screen:IView ):void {}
internal function goOut( screen:IView ):void {}

internal function doneIn():void { if (_inCallback != null) _inCallback(_name); }
internal function doneOut():void { if (_outCallback != null) _outCallback(_name); }

internal function addCallbacks( inCall:Function, outCall:Function ):void
{
_inCallback = inCall;
_outCallback = outCall;
}

public function destroy():void
{
_inCallback = _outCallback = null;
}
}
}
```

File 721: igw\com\ui\core\effects\EffectFactory.as

```
ï»¿package com.ui.core.effects
{
import flash.utils.Dictionary;

import org.shared.ObjectPool;

public class EffectFactory
{
private static var callbacksIN:Dictionary;
private static var callbacksOUT:Dictionary;

public
```

```
function EffectFactory() {}

public static function alphaEffect( start:Number, middle:Number, end:Number, timeIN:Number,
timeOUT:Number ):Effect
{
var effect:AlphaEffect = ObjectPool.get(AlphaEffect);
effect.init(start, middle, end, timeIN, timeOUT);
return effect;
}

public static function fullScreenFadeEffect( timeIN:Number, timeOUT:Number ):Effect
{
var effect:FullscreenFadeEffect = ObjectPool.get(FullscreenFadeEffect);
effect.init(timeIN, timeOUT);
return effect;
}

public static function genericMoveEffect( start:String, end:String, timeIN:Number,
timeOUT:Number, ei:Function = null, eo:Function = null ):Effect
{
var effect:GenericMoveEffect = ObjectPool.get(GenericMoveEffect);
effect.init(start, end, timeIN, timeOUT, ei, eo);
return effect;
}

public static function resizeEffect( callback:Function = null ):Effect
{
var effect:ResizeEffect = ObjectPool.get(ResizeEffect);
effect.init(callback);
return effect;
}

public static function simpleMoveEffect( start:String, end:String, timeIN:Number,
timeOUT:Number, ei:Function = null, eo:Function = null ):Effect
{
var effect:SimpleMoveEffect = ObjectPool.get(SimpleMoveEffect);
effect.init(start, end, timeIN, timeOUT, ei, eo);
return effect;
}

public static function simpleBackingEffect( a:Number, timeIN:Number, timeOUT:Number,
clickCallback:Function = null ):Effect
{
var effect:SimpleBackingEffect = ObjectPool.get(SimpleBackingEffect);
effect.init(a, timeIN, timeOUT, clickCallback);
return effect;
}

public static function scaleEffect( start:Number, end:Number, timeIN:Number, timeOUT:Number,
reposition:Boolean = false ):Effect
{
```

```
var effect:ScaleEffect = ObjectPool.get(ScaleEffect);
effect.init(start, end, timeIN, timeOUT, reposition);
return effect;
}

public static function repositionEffect( positionX:String, positionY:String, callback:Function = null,
screenX:Number = 0, screenY:Number = 0 ):Effect
{
var effect:StageRepositionEffect = ObjectPool.get(StageRepositionEffect);
effect.init(positionX, positionY, callback, screenX, screenY);
return effect;
}

public static function stageLetterboxEffect( timeIn:Number, timeOut:Number, isTop:Boolean,
boxHeight:int = 130, clickCallback:Function = null ):Effect
{
var effect:StageLetterboxEffect = ObjectPool.get(StageLetterboxEffect);
effect.init(timeIn, timeOut, isTop, boxHeight, clickCallback);
return effect;
}
}

}

-------------------------------------
File 722: igw\com\ui\core\effects\FullscreenFadeEffect.as
ï»¿package com.ui.core.effects
{
import com.Application;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Stage;
import flash.events.Event;

import org.greensock.TweenLite;
import org.parade.core.IView;

public class FullscreenFadeEffect extends Effect
{
public static const NAME:String = "FullscreenFadeEffect";

private static var COVER:Bitmap;

private var _stage:Stage;
private var _timeIN:Number;
private var _timeOUT:Number;

//adds a fade in effect to "screen"
public
```

```
function init( timeIN:Number, timeOUT:Number ):void
{
_name = NAME;
_timeIN = timeIN;
_timeOUT = timeOUT;

if (!COVER)
COVER = new Bitmap(new BitmapData(5, 5, false, 0));
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
_stage = Application.STAGE;
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);
resize();
COVER.alpha = 0;
Application.STAGE.addChild(COVER);
TweenLite.to(COVER, _timeIN, {alpha:1, onComplete:doneIn});
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
TweenLite.to(COVER, _timeOUT, {alpha:0, onComplete:doneOut});
}

protected function resize( e:Event = null ):void
{
COVER.width = _stage.stageWidth;
COVER.height = _stage.stageHeight;
}

override public function destroy():void
{
super.destroy();
if (_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}
TweenLite.killTweensOf(COVER);

if (COVER.parent)
COVER.parent.removeChild(COVER);
}
}

}
```

File 723: igw\com\ui\core\effects\GenericMoveEffect.as
ï»¿package com.ui.core.effects
{
import com.Application;

import flash.display.Stage;
import flash.events.Event;
import flash.geom.Point;
import flash.geom.Rectangle;

import org.as3commons.logging.level.ERROR;
import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.parade.core.IView;

public class GenericMoveEffect extends Effect
{
public static const NAME:String = "genericMoveEffect";
public static const LEFT:String = "left";
public static const RIGHT:String = "right";
public static const UP:String = "up";
public static const DOWN:String = "down";
public static const CENTER:String = "center";

private var _easeIn:Function;
private var _easeOut:Function;
private var _end:String;
private var _height:Number = 0;
private var _screen:IView;
private var _stage:Stage;
private var _start:String;
private var _state:int;
private var _timeIN:Number;
private var _timeOUT:Number;
private var _width:Number = 0;

public function init(start:String, end:String, timeIN:Number, timeOUT:Number, ei:Function = null,
eo:Function = null):void
{
_name = NAME;
_start = start;
_end = end;
_timeIN = timeIN;
_timeOUT = timeOUT;
_easeIn = (ei == null) ? Quad.easeOut : ei;
_easeOut = (eo == null) ? Quad.easeOut : eo;
}

override internal function goIn(screen:IView):void
{

```
super.goIn(screen);
_state = 0;
_stage = Application.STAGE;
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);
_screen = screen;
_height = (_height == 0) ? screen.height * screen.scaleY : _height;
_width = (_width == 0) ? screen.width * screen.scaleX : _width;
var start:Point = position(screen, _start);
var end:Point = position(screen, CENTER);
screen.x = start.x;
screen.y = start.y;

TweenLite.to(screen, _timeIN, {x:end.x, y:end.y, onComplete:doneIn, ease:_easeIn,
overwrite:false});
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
_state = 1;
var end:Point = position(screen, _end);
TweenLite.to(screen, _timeOUT, {x:end.x, y:end.y, onComplete:doneOut, ease:_easeOut,
overwrite:false});
}

private function position( screen:IView, pos:String ):Point
{
try
{
_height = (screen) ? screen.height * screen.scaleY : _height;
_width = (screen) ? screen.width * screen.scaleX : _width;
} catch ( e:Error )
{
_height = _height;
_width = _width;
}
var rect:Rectangle = screen.bounds;
var point:Point = new Point(_stage.stageWidth / 2, _stage.stageHeight / 2);
var center:Point = new Point(rect.left + (_width / 2), rect.top + (_height / 2));
var xDiff:Number = center.x - screen.x;
var yDiff:Number = center.y - screen.y;

switch (pos)
{
case LEFT:
point.x = 0 - Math.abs(rect.right - screen.x);
point.y -= yDiff;
break;
case
```

```
RIGHT:
point.x = _stage.stageWidth + Math.abs(screen.x - rect.left)
point.y -= yDiff;
break;
case UP:
point.y = 0 - Math.abs(rect.bottom - screen.y);
point.x -= xDiff;
break;
case DOWN:
point.y = _stage.stageHeight + Math.abs(screen.y - rect.top);
point.x -= xDiff;
break;
case CENTER:
point.x -= xDiff;
point.y -= yDiff;
break;
}
return point;
}

protected function resize( e:Event = null ):void
{
var p:Point = (_state == 0) ? position(_screen, CENTER) : position(_screen, _end);
TweenLite.killTweensOf(_screen, false, {x:true, y:true});
TweenLite.to(_screen, (_state == 0) ? _timeIN : _timeOUT, {x:p.x, y:p.y, onComplete:(_state ==
0) ? doneIn : doneOut, ease:(_state == 0) ? _easeIn : _easeOut, overwrite:false});
}

public function set height( v:Number ):void { _height = v; }
public function set width( v:Number ):void { _width = v; }

override public function destroy():void
{
super.destroy();
if (_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}
_screen = null;
_width = _height = 0;
_easeIn = null;
_easeOut = null;
}
}

}

-----------------------------------
File 724: igw\com\ui\core\effects\ResizeEffect.as
package
```

```
com.ui.core.effects
{
import com.Application;

import flash.display.Stage;
import flash.events.Event;

import org.parade.core.IView;
import org.parade.util.DeviceMetrics;

public class ResizeEffect extends Effect
{
public static const NAME:String = "stageResizeEffect";

private var _callback:Function;
private var _screen:IView;
private var _stage:Stage;

public function init( callback:Function = null ):void
{
_callback = callback;
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
_screen = screen;
_stage = Application.STAGE;
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);
resize();
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
_stage.removeEventListener(Event.RESIZE, resize);
doneOut();
}

protected function resize( e:Event = null ):void
{
var h:Number = Math.max(_screen.height + 20, DeviceMetrics.HEIGHT_PIXELS);
var w:Number = Math.max(_screen.width + 20, DeviceMetrics.WIDTH_PIXELS);
var scaleH:Number = DeviceMetrics.HEIGHT_PIXELS / h;
var scaleW:Number = DeviceMetrics.WIDTH_PIXELS / w;
_screen.scaleX = _screen.scaleY = Math.min(scaleH, scaleW);
if (_callback != null)
_callback();
doneIn();
}
```

```
override public function destroy():void
{
if (_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}
_screen = null;
_callback = null;
}
}
}

-----------------------------------
File 725: igw\com\ui\core\effects\ScaleEffect.as
package com.ui.core.effects
{
import org.greensock.TweenLite;
import org.parade.core.IView;

public class ScaleEffect extends Effect
{
public static const NAME:String = "scaleEffect";

private var _start:Number;
private var _end:Number;
private var _reposition:Boolean;
private var _timeIN:Number;
private var _timeOUT:Number;

public function init( start:Number, end:Number, timeIN:Number, timeOUT:Number,
reposition:Boolean = false ):void
{
_name = NAME;
_start = start;
_end = end;
_reposition = reposition;
_timeIN = timeIN;
_timeOUT = timeOUT;
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
if (screen.scaleX != _end)
{
if (_reposition)
reposition(screen, true);
if
```

```
(_timeIN > 0)
TweenLite.to(screen, _timeIN, {scaleX:_end, scaleY:_end, onComplete:doneIn,
overwrite:false});
else
{
screen.scaleX = _end;
screen.scaleY = _end;
doneIn();
}
}
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
if (screen.scaleX != _start)
{
TweenLite.killTweensOf(screen, false, {scaleX:true, scaleY:true});
if (_reposition)
reposition(screen, false);
if (_timeOUT > 0)
TweenLite.to(screen, _timeOUT, {scaleX:_start, scaleY:_start, onComplete:doneOut,
overwrite:false});
else
{
screen.scaleX = _start;
screen.scaleY = _start;
doneOut();
}
}
}

protected function reposition( screen:IView, start:Boolean ):void
{
var ds:Number = screen.scaleX;
screen.scaleX = screen.scaleY = _start;
var diff:Number = _end - _start;
var dx:Number = screen.x - Math.round(screen.width * diff / 2 * ((start) ? 1 : -1));
var dy:Number = screen.y - Math.round(screen.height * diff / 2 * ((start) ? 1 : -1));
screen.scaleX = screen.scaleY = ds;
var time:Number = (start) ? _timeIN : _timeOUT;
if (time > 0)
TweenLite.to(screen, time, {x:dx, y:dy, overwrite:false});
else
{
screen.x = dx;
screen.y = dy;
}
}
}

}
```

File 726: igw\com\ui\core\effects\SimpleBackingEffect.as

```
package com.ui.core.effects
{
import com.Application;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.display.Stage;
import flash.events.Event;
import flash.events.MouseEvent;

import org.greensock.TweenLite;
import org.parade.core.IView;

public class SimpleBackingEffect extends Effect
{
public static const NAME:String = "simpleBackingEffect";

private static var BACKING:Bitmap;
private static var INTERACTIVE_OBJECT:Sprite;
private static var APPLY_TO:Array;
private static var CLICK_CALLBACKS:Array;

private var _alpha:Number;
private var _timeIN:Number;
private var _timeOUT:Number;
private var _stage:Stage;

public function init( a:Number, timeIN:Number, timeOUT:Number, clickCallback:Function = null
):void
{
_name = NAME;
if (!BACKING)
{
BACKING = new Bitmap(new BitmapData(5, 5, false, 0));
INTERACTIVE_OBJECT = new Sprite();
INTERACTIVE_OBJECT.addChild(BACKING);
BACKING.alpha = 0;
APPLY_TO = [];
CLICK_CALLBACKS = [];
}

INTERACTIVE_OBJECT.addEventListener(MouseEvent.CLICK, onMouseEvent, false, 0, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_UP, onMouseEvent, false, 1,
true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_DOWN, onMouseEvent,
false,
```

```
1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_WHEEL, onMouseEvent,
false, 1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_MOVE, onMouseEvent,
false, 1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.ROLL_OUT, onMouseEvent, false, 0,
true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.ROLL_OVER, onMouseEvent, false, 0,
true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.RIGHT_CLICK, onMouseEvent, false,
1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.RIGHT_MOUSE_DOWN,
onMouseEvent, false, 1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.RIGHT_MOUSE_UP, onMouseEvent,
false, 1, true);

_alpha = a;
CLICK_CALLBACKS.push(clickCallback);
_timeIN = timeIN;
_timeOUT = timeOUT;
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
_stage = Application.STAGE
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);
resize();
APPLY_TO.push(screen);

var s:* = screen;
s.parent.addChild(INTERACTIVE_OBJECT);
s.parent.swapChildren(INTERACTIVE_OBJECT, screen);

if (BACKING.alpha < _alpha || BACKING.alpha > _alpha)
TweenLite.to(BACKING, _timeIN, {alpha:_alpha, onComplete:doneIn});
else
doneIn();
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
var index:int = APPLY_TO.indexOf(screen);
if (index > -1)
APPLY_TO.splice(index, 1);
if (CLICK_CALLBACKS.length > 0)
CLICK_CALLBACKS.pop();
if (APPLY_TO.length > 0)
{
```

```
screen = APPLY_TO[APPLY_TO.length - 1];
var s:* = screen;
s.parent.addChild(INTERACTIVE_OBJECT);
s.parent.swapChildren(INTERACTIVE_OBJECT, screen);
doneOut();
} else
{
_stage.removeEventListener(Event.RESIZE, resize);
TweenLite.to(BACKING, _timeOUT, {alpha:0, onComplete:doneOut});
}
}

private function onMouseEvent( e:MouseEvent ):void
{
e.stopImmediatePropagation();
if (e.type == MouseEvent.CLICK && CLICK_CALLBACKS.length > 0 &&
CLICK_CALLBACKS[CLICK_CALLBACKS.length - 1] != null)
CLICK_CALLBACKS[CLICK_CALLBACKS.length - 1]();
}

protected function resize( e:Event = null ):void
{
BACKING.width = _stage.stageWidth;
BACKING.height = _stage.stageHeight;
}

override public function destroy():void
{
if (APPLY_TO.length == 0)
{
if (_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}
if (INTERACTIVE_OBJECT.parent)
{
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.CLICK, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_UP, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_DOWN, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_WHEEL, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.RIGHT_CLICK, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.RIGHT_MOUSE_DOWN,
onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.RIGHT_MOUSE_UP,
onMouseEvent);
INTERACTIVE_OBJECT.parent.removeChild(INTERACTIVE_OBJECT);
}
TweenLite.killTweensOf(BACKING);
```

```
}
}
}

}

-------------------------------------
File 727: igw\com\ui\core\effects\SimpleMoveEffect.as
package com.ui.core.effects
{
import com.Application;

import flash.display.Stage;
import flash.events.Event;
import flash.geom.Point;
import flash.geom.Rectangle;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.parade.core.IView;

public class SimpleMoveEffect extends Effect
{
public static const NAME:String = "simpleMoveEffect";
public static const LEFT:String = "left";
public static const RIGHT:String = "right";
public static const UP:String = "up";
public static const DOWN:String = "down";

private var _easeIn:Function;
private var _easeOut:Function;
private var _end:String;
private var _height:Number = 0;
private var _screen:IView;
private var _stage:Stage;
private var _start:String;
private var _state:int;
private var _timeIN:Number;
private var _timeOUT:Number;
private var _width:Number = 0;

public function init( start:String, end:String, timeIN:Number, timeOUT:Number, ei:Function = null,
eo:Function = null ):void
{
_start = start;
_end = end;
_timeIN = timeIN;
_timeOUT = timeOUT;
_easeIn = (ei == null) ? Quad.easeOut : ei;
_easeOut
```

```
= (eo == null) ? Quad.easeOut : eo;
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
_state = 0;
_stage = Application.STAGE;
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);
_screen = screen;
_height = (_height == 0) ? screen.height : _height;
_width = (_width == 0) ? screen.width : _width;
var start:Point = position(screen, _start);

TweenLite.from(screen, _timeIN, {x:start.x, y:start.y, onComplete:doneIn, ease:_easeIn,
overwrite:false});
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
_state = 1;
var end:Point = position(screen, _end);
TweenLite.to(screen, _timeOUT, {x:end.x, y:end.y, onComplete:doneOut, ease:_easeOut,
overwrite:false});
}

private function position( screen:IView, pos:String ):Point
{
var rect:Rectangle = screen.bounds;
var point:Point = new Point(screen.x, screen.y);

switch (pos)
{
case LEFT:
point.x = 0 - Math.abs(rect.right - screen.x);
break;
case RIGHT:
point.x = _stage.stageWidth + Math.abs(screen.x - rect.left)
break;
case UP:
point.y = 0 - Math.abs(rect.bottom - screen.y);
break;
case DOWN:
point.y = _stage.stageHeight + Math.abs(screen.y - rect.top);
break;
}

return point;
}
```

```
protected function resize( e:Event = null ):void
{
/*var p:Point = (_state == 0) ? position(_screen, CENTER) : position(_screen, _end);
TweenLite.killTweensOf(_screen, false, {x:true, y:true});
TweenLite.to(_screen, (_state == 0) ? _timeIN : _timeOUT, {x:p.x, y:p.y, onComplete:(_state ==
0) ? doneIN : doneOUT, ease:(_state == 0) ? _easeIn : _easeOut, overwrite:false});*/
}

public function set height( v:Number ):void { _height = v; }
public function set width( v:Number ):void { _width = v; }

override public function destroy():void
{
if (_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}
_screen = null;
_width = _height = 0;
_easeIn = null;
_easeOut = null;
}
}
}

-------------------------------------
File 728: igw\com\ui\core\effects\StageLetterboxEffect.as
package com.ui.core.effects
{
import com.Application;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.display.Stage;
import flash.events.Event;
import flash.events.MouseEvent;

import org.greensock.TweenLite;
import org.parade.core.IView;

public class StageLetterboxEffect extends Effect
{
public
```

```
static const NAME:String = "stageLetterboxEffect";

private var LETTERBOX:Bitmap;
private var INTERACTIVE_OBJECT:Sprite;

private var _clickCallback:Function;
private var _isTop:Boolean;
private var _screen:IView;
private var _stage:Stage;
private var _timeIn:Number;
private var _timeOut:Number;
private var _titleBar:ScaleBitmap;

public function init( timeIn:Number, timeOut:Number, isTop:Boolean, boxHeight:int = 130,
clickCallback:Function = null ):void
{
_clickCallback = clickCallback;
_name = NAME;
_timeIn = timeIn;
_timeOut = timeOut;

LETTERBOX = new Bitmap(new BitmapData(5, 5, false, 0));
LETTERBOX.height = boxHeight;

INTERACTIVE_OBJECT = new Sprite();
INTERACTIVE_OBJECT.addChild(LETTERBOX);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.CLICK, onMouseEvent, false, 0, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_UP, onMouseEvent, false, 1,
true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_DOWN, onMouseEvent,
false, 1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.MOUSE_WHEEL, onMouseEvent,
false, 1, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.RIGHT_CLICK, onMouseEvent, false,
0, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.RIGHT_MOUSE_DOWN,
onMouseEvent, false, 0, true);
INTERACTIVE_OBJECT.addEventListener(MouseEvent.RIGHT_MOUSE_UP, onMouseEvent,
false, 0, true);

if (!isTop)
{
_titleBar = UIFactory.getScaleBitmap(PanelEnum.HEADER);
INTERACTIVE_OBJECT.addChild(_titleBar);
}
_isTop = isTop;
}

override internal function goIn( screen:IView ):void
{
_stage
```

```
= Application.STAGE
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);

_screen = screen;
View(_screen).parent.addChild(INTERACTIVE_OBJECT);
View(_screen).parent.swapChildren(INTERACTIVE_OBJECT, View(_screen));
resize();

if (_timeIn > 0)
{
INTERACTIVE_OBJECT.alpha = 0;
TweenLite.to(INTERACTIVE_OBJECT, _timeIn, {alpha:1});
}

doneIn();
}

override internal function goOut( screen:IView ):void
{
doneOut();

if (_timeOut > 0)
TweenLite.to(INTERACTIVE_OBJECT, _timeOut, {alpha:0});
}

protected function resize( e:Event = null ):void
{
LETTERBOX.width = _stage.stageWidth;
INTERACTIVE_OBJECT.scaleY = _screen.scaleX;

if (!_isTop)
{
_titleBar.width = LETTERBOX.width;
_titleBar.height = 30;
_titleBar.x = _titleBar.y = 0;
INTERACTIVE_OBJECT.y = _stage.stageHeight - LETTERBOX.height * _screen.scaleX;
}
}

private function onMouseEvent( e:MouseEvent ):void
{
e.stopImmediatePropagation();
if (e.type == MouseEvent.CLICK && _clickCallback != null)
_clickCallback();
}

override public function destroy():void
{
_clickCallback = null;
if
```

```
(_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}

INTERACTIVE_OBJECT.removeEventListener(MouseEvent.CLICK, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_UP, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_DOWN, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.MOUSE_WHEEL, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.RIGHT_CLICK, onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.RIGHT_MOUSE_DOWN,
onMouseEvent);
INTERACTIVE_OBJECT.removeEventListener(MouseEvent.RIGHT_MOUSE_UP,
onMouseEvent);

TweenLite.killTweensOf(INTERACTIVE_OBJECT);

if (INTERACTIVE_OBJECT.parent)
INTERACTIVE_OBJECT.parent.removeChild(INTERACTIVE_OBJECT);

while (INTERACTIVE_OBJECT.numChildren > 0)
INTERACTIVE_OBJECT.removeChildAt(0);
INTERACTIVE_OBJECT = null;

if (_titleBar)
_titleBar = UIFactory.destroyPanel(_titleBar);
_screen = null;
}
}
}

------------------------------------
File 729: igw\com\ui\core\effects\StageRepositionEffect.as
package com.ui.core.effects
{
import com.Application;
import com.enum.PositionEnum;

import flash.display.Stage;
import flash.events.Event;
import flash.utils.setTimeout;

import org.parade.core.IView;

public class StageRepositionEffect extends Effect
{
public static const NAME:String = "stageRepositionEffect";

private var _callback:Function;
private
```

```
var _offsetX:Number;
private var _offsetY:Number;
private var _positionX:String;
private var _positionY:String;
private var _screen:IView;
private var _stage:Stage;

public function init( positionX:String, positionY:String, callback:Function = null, screenX:Number
= 0, screenY:Number = 0 ):void
{
_callback = callback;
_positionX = positionX;
_positionY = positionY;
_offsetX = screenX;
_offsetY = screenY;
}

override internal function goIn( screen:IView ):void
{
super.goIn(screen);
_screen = screen;
_stage = Application.STAGE;
_stage.removeEventListener(Event.RESIZE, resize);
_stage.addEventListener(Event.RESIZE, resize, false, 0, true);
setTimeout(getOffsets, 50);
doneIn();
}

override internal function goOut( screen:IView ):void
{
super.goOut(screen);
_stage.removeEventListener(Event.RESIZE, resize);
doneOut();
}

protected function getOffsets():void
{
// @todo TK 1/31/13 Temporary hack fix to guard against null _screen
if (!_screen)
return;

var x:Number = (_offsetX != 0) ? _offsetX : _screen.x;
var y:Number = (_offsetY != 0) ? _offsetY : _screen.y;
switch (_positionX)
{
case PositionEnum.LEFT:
_offsetX = x;
break;
case PositionEnum.CENTER:
_offsetX = x - (_stage.stageWidth / 2);
break;
```

```
default:
_offsetX = _stage.stageWidth - x;
break;
}

switch (_positionY)
{
case PositionEnum.TOP:
_offsetY = y;
break;
case PositionEnum.CENTER:
_offsetY = y - (_stage.stageHeight / 2);
break;
default:
_offsetY = _stage.stageHeight - y;
break;
}
}

protected function resize( e:Event = null ):void
{
switch (_positionX)
{
case PositionEnum.LEFT:
_screen.x = 0 + _offsetX;
break;
case PositionEnum.CENTER:
_screen.x = (_stage.stageWidth / 2) + _offsetX;
break;
default:
_screen.x = _stage.stageWidth - _offsetX;
break;
}

switch (_positionY)
{
case PositionEnum.TOP:
_screen.y = 0 + _offsetY;
break;
case PositionEnum.CENTER:
_screen.y = (_stage.stageHeight / 2) + _offsetY;
break;
default:
_screen.y = _stage.stageHeight - _offsetY;
break;
}
if (_callback != null)
_callback();
doneIn();
}
```

```
override public function destroy():void
{
if (_stage)
{
_stage.removeEventListener(Event.RESIZE, resize);
_stage = null;
}
_screen = null;
_callback = null;
}
}
}
```

File 730: igw\com\ui\core\effects\ViewEffects.as

```
package com.ui.core.effects
{
import org.osflash.signals.Signal;
import org.parade.core.IView;
import org.shared.ObjectPool;

public class ViewEffects
{
private var _doneIn:Boolean = false;
private var _doneOut:Boolean = false;
private var _effects:Vector.<Effect> = new Vector.<Effect>;
private var _finishCount:int;
private var _onEffectIn:Signal = new Signal();
private var _onEffectOut:Signal = new Signal();

public function addEffect( effect:Effect ):void
{
effect.addCallbacks(doneIn, doneOut);
_effects.push(effect);
}

public function effectsIn( screen:IView ):void
{
_finishCount = 0;
for (var i:int = 0; i < _effects.length; i++)
{
_effects[i].goIn(screen);
}
if (_effects.length == 0)
doneIn('');
}

public function effectsOut( screen:IView ):void
{
```

```
_finishCount = 0;
for (var i:int = 0; i < _effects.length; i++)
{
_effects[i].goOut(screen);
}
if (_effects.length == 0)
doneOut('');
}

private function doneIn( name:String ):void
{
_doneIn = true;
_finishCount++;
if (_finishCount >= _effects.length)
_onEffectIn.dispatch();
}

private function doneOut( name:String ):void
{
_doneOut = true;
_finishCount++;
if (_finishCount >= _effects.length)
_onEffectOut.dispatch();
}

public function addInListener( listener:Function ):void { _onEffectIn.add(listener); }
public function addOutListener( listener:Function ):void { _onEffectOut.add(listener); }
public function removeInListener( listener:Function ):void { _onEffectIn.remove(listener); }
public function removeOutListener( listener:Function ):void { _onEffectOut.remove(listener); }

public function get isDoneIn():Boolean { return _doneIn; }
public function get isDoneOut():Boolean { return _doneOut; }
public function get numEffects():int { return _effects.length; }

public function destroy():void
{
for (var i:int = 0; i < _effects.length; i++)
{
ObjectPool.give(_effects[i]);
}
_doneIn = _doneOut = false;
_effects.length = 0;
_onEffectIn.removeAll();
_onEffectOut.removeAll();
}
}
}

-----------------------------------
File
```

731: igw\com\ui\hud\battle\BattleBaseView.as
```
package com.ui.hud.battle
{
import com.presenter.battle.IBattlePresenter;
import com.ui.core.View;

import org.parade.enum.ViewEnum;

public class BattleBaseView extends View
{
[PostConstruct]
override public function init():void
{
super.init();
presenter.addCleanupListener(destroy);
}

[Inject]
public function set presenter( value:IBattlePresenter ):void { _presenter = value; }
public function get presenter():IBattlePresenter { return IBattlePresenter(_presenter); }

override public function get type():String { return ViewEnum.UI; }

override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
presenter.removeCleanupListener(destroy);
super.destroy();
}
}
}
```

File 732: igw\com\ui\hud\battle\BattleShipSelectionView.as
```
ï»¿package com.ui.hud.battle
{
import com.Application;
import com.enum.PositionEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.PanelEnum;
import com.game.entity.components.battle.Health;
import com.game.entity.systems.interact.controls.ControlledEntity;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.player.CurrentUser;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.effects.EffectFactory;
import
```

```
com.ui.modal.dock.ShipIcon;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.events.KeyboardEvent;
import flash.ui.Keyboard;

import org.ash.core.Entity;
import org.greensock.TweenLite;
import org.parade.util.DeviceMetrics;

public class BattleShipSelectionView extends BattleBaseView
{
private var _bg:Sprite;
private var _fleet:FleetVO;
private var _fleetFrame:Bitmap;
private var _fleetName:Label;
private var _minimizeButton:BitmapButton;
private var _maximizeButton:BitmapButton;
private var _selectAllShipsButton:BitmapButton;
private var _selectedFleetShips:Vector.<ShipIcon>;
private var _ships:Dictionary;
private var _windowState:int;

private const MIN_X_POS:Number = 271;
private const MIN_Y_POS:Number = 467;
private const MAXIMIZED:Number = 1;
private const MINIMIZED:Number = 0;

[PostConstruct]
override public function init():void
{
_windowState = MAXIMIZED;
_fleet = presenter.getSelectedFleet();
if (_fleet == null)
return;

_ships = new Dictionary();

super.init();

_selectedFleetShips = new Vector.<ShipIcon>;

_bg = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_DOUBLE_NOTCHED_ARROWS,
PanelEnum.HEADER_NOTCHED, 225, 156, 32, 0, 0, _fleet.name);
_bg.mouseEnabled = false;

_fleetFrame
```

```
= UIFactory.getBitmap('SectorFleetSelectionBGBMD');
_fleetFrame.x = _bg.x + (_bg.width - _fleetFrame.width) * 0.5;
_fleetFrame.y = _bg.y + (_bg.height - _fleetFrame.height) * 0.5 + 15;

_fleetName = new Label(22, 0xf0f0f0, 200, 30);
_fleetName.allCaps = true;
_fleetName.useLocalization = false;
_fleetName.align = TextFormatAlign.LEFT;
_fleetName.x = 3;
_fleetName.y = 3;

_minimizeButton = UIFactory.getButton(ButtonEnum.ICON_WINDOW_MIN, 0, 0, 205, 13);
_maximizeButton = UIFactory.getButton(ButtonEnum.ICON_WINDOW_FULL, 0, 0, 205, 13);
_maximizeButton.visible = false;

_selectAllShipsButton = UIFactory.getButton(ButtonEnum.BLUE_A, 225, 46, 0, 0, "Select All");
/*if(CONFIG::IS_MOBILE){
_selectAllShipsButton.scaleX = _selectAllShipsButton.scaleY = 2;
}*/
_selectAllShipsButton.y = -(_selectAllShipsButton.height + 4);
_selectAllShipsButton.x = _bg.x;
_selectAllShipsButton.addEventListener(MouseEvent.CLICK, _onSelectAllShipsClicked);
addListener(_minimizeButton, MouseEvent.CLICK, onMinimize);
addListener(_maximizeButton, MouseEvent.CLICK, onMaximize);

addChild(_bg);

addChild(_minimizeButton);
addChild(_maximizeButton);
addChild(_fleetFrame);
addChild(_fleetName);
addChild(_selectAllShipsButton);

presenter.addListenerVitalPercentUpdates(onHealthUpdated);
presenter.addListenerBattleEntitiesControlledUpdated(BattleEntitiesControlledUpdated);

onStageResize();
setUpShips();

addHitArea();
addEffects();
effectsIN();

visible = !presenter.inFTE;
}

private function _onSelectAllShipsClicked(e:MouseEvent):void{
stage.dispatchEvent(new KeyboardEvent(KeyboardEvent.KEY_DOWN, true, false, 0,
Keyboard.Q));
stage.dispatchEvent(new KeyboardEvent(KeyboardEvent.KEY_UP, true, false, 0, Keyboard.Q));

}
```

```
private function setUpShips():void
{
var image:ShipIcon;
var xPos:Number;
var yPos:Number;
var currentShip:ShipVO;
var currentEntity:Entity;
//68.5
//143
for (var i:uint = 0; i < 6; ++i)
{
image = new ShipIcon();
image.scale(0.37, 0.37);
image.index = i;

switch (i)
{
case 0:
xPos = _fleetFrame.x + 47;
yPos = _fleetFrame.y;
break;
case 1:
xPos = _fleetFrame.x + 1;
yPos = _fleetFrame.y + 26;
break;
case 2:
xPos = _fleetFrame.x + 91;
yPos = _fleetFrame.y + 26;
break;
case 3:
xPos = _fleetFrame.x + 1;
yPos = _fleetFrame.y + 77;
break;
case 4:
xPos = _fleetFrame.x + 91;
yPos = _fleetFrame.y + 77;
break;
case 5:
xPos = _fleetFrame.x + 45;
yPos = _fleetFrame.y + 105;
break;
}

image.x = xPos;
image.y = yPos;
addChild(image);

if (_fleet != null)
currentShip
```

```
= _fleet.ships[i];

if (currentShip)
{
currentEntity = presenter.getEntity(currentShip.id);
image.onLoadShipImage.add(presenter.loadMiniIconFromEntityData);
image.setShip(currentShip, null, _fleet);
if (currentEntity)
{
var health:Health = currentEntity.get(Health);
image.setBarValue(1 - health.percent);
image.selectable = true;
} else
image.setBarValue(1);

addListener(image, MouseEvent.CLICK, onShipClicked);

_ships[currentShip.id] = image;
}
_selectedFleetShips.push(image);
}
}

private function onShipClicked( e:MouseEvent ):void
{
var icon:ShipIcon;
if (e.target is ShipIcon)
icon = ShipIcon(e.target);
else if (e.target.parent is ShipIcon)
icon = ShipIcon(e.target.parent);

if (icon)
{
var ship:ShipVO = icon.ship;
if (ship)
presenter.selectOwnedShipById(ship.id)
}
e.stopPropagation();
}

private function onHealthUpdated( playerId:String, percent:Number ):void
{
if (playerId == CurrentUser.id)
{
var selectedShipIcon:ShipIcon;
var selectedShip:ShipVO;
for each (var ship:ShipIcon in _ships)
{
selectedShip = ship.ship;

if
```

```
(selectedShip)
{
var entity:Entity = presenter.getShip(selectedShip.id);
if (entity)
{
var health:Health = entity.get(Health);
if (health)
ship.setBarValue(1 - health.percent);
}
}
}
}
}

private function onMinimize( e:MouseEvent ):void
{
_windowState = MINIMIZED;
_minimizeButton.visible = false;
_maximizeButton.visible = true;
TweenLite.to(this, .2, {y:DeviceMetrics.HEIGHT_PIXELS - 32});
e.stopPropagation();
}

private function onMaximize( e:MouseEvent ):void
{
_windowState = MAXIMIZED;
_minimizeButton.visible = true;
_maximizeButton.visible = false;
TweenLite.to(this, .2, {y:DeviceMetrics.HEIGHT_PIXELS - (_bg.height * Application.SCALE)});
e.stopPropagation();
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.RIGHT,
PositionEnum.BOTTOM, onStageResize, x, y));
}

private function onStageResize():void
{
this.scaleX = this.scaleY = Application.SCALE;
TweenLite.killTweensOf(this);
var yPos:Number;
switch (_windowState)
{
case MAXIMIZED:
yPos = DeviceMetrics.HEIGHT_PIXELS - height;
break;
case MINIMIZED:
yPos = DeviceMetrics.HEIGHT_PIXELS - (32 * Application.SCALE);
break;
```

```
}
y = (yPos < MIN_Y_POS) ? MIN_Y_POS : yPos;
x = (DeviceMetrics.WIDTH_PIXELS - width < MIN_X_POS) ? MIN_X_POS :
DeviceMetrics.WIDTH_PIXELS - width;
}

private function BattleEntitiesControlledUpdated( v:Vector.<ControlledEntity> ):void
{
var len:uint = v.length;
var currentEntity:ControlledEntity;
var controlledList:Dictionary = new Dictionary;
for (var i:uint = 0; i < len; ++i)
controlledList[v[i].entity.id] = "";

for each (var ship:ShipIcon in _ships)
{
if (ship.id in controlledList)
{
ship.selectable = true;
ship.selected = true;
ship.selectable = false;
} else
{
ship.selectable = true;
ship.selected = false;
}
}
}

override public function get height():Number { return _bg.height * Application.SCALE; }
override public function get width():Number { return _bg.width * Application.SCALE; }

override public function destroy():void
{
_selectAllShipsButton.removeEventListener(MouseEvent.CLICK, _onSelectAllShipsClicked);
presenter.removeListenerVitalPercentUpdates(onHealthUpdated);
var len:uint = _selectedFleetShips.length;
var selectedShipIcon:ShipIcon;
for (var i:uint = 0; i < len; ++i)
{
selectedShipIcon = _selectedFleetShips[i];

if (selectedShipIcon.ship)
removeListener(selectedShipIcon, MouseEvent.CLICK, onShipClicked);

selectedShipIcon.destroy();
selectedShipIcon = null;
}
_selectedFleetShips.length = 0;

_bg
```

```
= null;
_fleetFrame = null;

if (_fleetName)
_fleetName.destroy();

_fleetName = null;

super.destroy();
}
}
}

-----------------------------------
File 733: igw\com\ui\hud\battle\BattleUserView.as
ï»¿package com.ui.hud.battle
{
import com.Application;
import com.enum.FactionEnum;
import com.enum.PositionEnum;
import com.enum.ToastEnum;
import com.enum.ui.PanelEnum;
import com.model.fleet.FleetVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.label.Label;
import com.ui.core.effects.EffectFactory;

import flash.display.Sprite;
import flash.events.Event;
import flash.events.TimerEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.Timer;

public class BattleUserView extends BattleBaseView
{
private var _battleTimer:Timer;

private var _hitArea:Sprite;
private var _leftAttach:Sprite;
private var _rightAttach:Sprite;

private var _timeContainer:ScaleBitmap;

private var _playerFrames:Dictionary;

private
```

```
var _overrideFaction:String;

private var _timeRemainingHeader:Label;
private var _timeRemaining:Label;

private var _bubblePrototype:IPrototype;

private var _isPlayerInvolved:Boolean;
private var _isBaseCombat:Boolean;
private var _isPlayersBase:Boolean;
private var _isInstancedMission:Boolean;

private var _firstBubbleThreshold:Number;
private var _secondBubbleThreshold:Number;
private var _thirdBubbleThreshold:Number;
private var _tempTime:Number;

private var _timeRemainingText:String = 'CodeString.BattleUserView.TimeRemaining'; //TIME
REMAINING

private var _firstThresholdTitle:String = 'CodeString.Toast.BattleBubbleFirstThresholdTitle';
//Defenses Breached!
private var _firstThresholdBody:String = 'CodeString.Toast.BattleBubbleFirstThresholdBody';
//Your opponent gained 18 hours of Base Protection

private var _secondThresholdTitle:String =
'CodeString.Toast.BattleBubbleSecondThresholdTitle'; //Base Crippled!
private var _secondThresholdBody:String =
'CodeString.Toast.BattleBubbleSecondThresholdBody'; //Your opponent gained 24 hours of
Base Protection

private var _thirdThresholdTitle:String = 'CodeString.Toast.BattleBubbleThirdThresholdTitle';
//Total Devastation!
private var _thirdThresholdBody:String = 'CodeString.Toast.BattleBubbleThirdThresholdBody';
//Your opponent gained 36 hours of Base Protection

private const MIN_WIDTH:Number = 732;

[PostConstruct]
override public function init():void
{
super.init();
presenter.addListenerVitalPercentUpdates(onHealthUpdated);
presenter.addStartListener(onBattleStart);

_leftAttach = new Sprite();
_leftAttach.x = 2;

_rightAttach = new Sprite();
_rightAttach.x = 375;
_leftAttach.y
```

```
= _rightAttach.y = 3;

_battleTimer = new Timer(1000, 0);
addListener(_battleTimer, TimerEvent.TIMER, onBattleTimer);

_isPlayerInvolved = presenter.isPlayerInCombat(CurrentUser.id);
_isBaseCombat = presenter.isBaseCombat;
_isPlayersBase = presenter.isPlayerBaseOwner(CurrentUser.id);
_isInstancedMission = presenter.isInstancedMission();

if (_isBaseCombat)
{
_bubblePrototype = presenter.getStoreItemPrototypeByName('Bubble_28d');
_firstBubbleThreshold =
presenter.getConstantPrototypeValueByName('protectionLowDamageThreshold');
_secondBubbleThreshold =
presenter.getConstantPrototypeValueByName('protectionMediumDamageThreshold');
_thirdBubbleThreshold =
presenter.getConstantPrototypeValueByName('protectionHighDamageThreshold');
}

_playerFrames = new Dictionary();
_overrideFaction = '';
var participantIDs:Vector.<String> = presenter.participants;
var len:uint = participantIDs.length;
var participants:Vector.<PlayerVO> = new Vector.<PlayerVO>();
var i:uint;
var currentPlayer:PlayerVO;
for (; i < len; ++i)
{

currentPlayer = presenter.getPlayer(presenter.participants[i]);
participants.push(currentPlayer);
if(_isInstancedMission)
{
if(i == 0)
_overrideFaction = currentPlayer.faction;
}
else if (_overrideFaction == '' && !_isPlayerInvolved && currentPlayer.faction !=
FactionEnum.IMPERIUM && currentPlayer.isNPC == false)
_overrideFaction = currentPlayer.faction;
}

len = participants.length;
for (i = 0; i < len; ++i)
addParticipantFrame(participants[i], (_isInstancedMission) ? _overrideFaction :
((_isPlayerInvolved) ? CurrentUser.faction : _overrideFaction));

layoutParticipants();

_timeContainer
```

```
= UIFactory.getScaleBitmap(PanelEnum.CONTAINER_NOTCHED_LEFT_SMALL);
_timeContainer.width = 90;
_timeContainer.height = 75;
_timeContainer.x = 281;
_timeContainer.y = 3;

_timeRemainingHeader = new Label(24, 0xf0f0f0, 88, 30);
_timeRemainingHeader.x = _timeContainer.x;
_timeRemainingHeader.y = _timeContainer.y + 15;
_timeRemainingHeader.align = TextFormatAlign.CENTER;
_timeRemainingHeader.text = _timeRemainingText;

_timeRemaining = new Label(34, 0xf0f0f0, 88, 40);
_timeRemaining.useLocalization = false;
_timeRemaining.x = _timeRemainingHeader.x;
_timeRemaining.y = _timeRemainingHeader.y + _timeRemainingHeader.textHeight;
_timeRemaining.align = TextFormatAlign.CENTER;
_timeRemaining.constrictTextToSize = false;
_timeRemaining.text = '';z

addChild(_leftAttach)
addChild(_rightAttach);
addChild(_timeContainer);
addChild(_timeRemainingHeader);
addChild(_timeRemaining);

addHitArea();
addEffects();
effectsIN();
onStageResize();

visible = !presenter.inFTE;

presenter.addListenerOnParticipantsAdded(onParticipantsAdded);
}

private function onHealthUpdated( id:String, percent:Number ):void
{
if (_playerFrames.hasOwnProperty(id))
{
_isPlayerInvolved = presenter.isPlayerInCombat(CurrentUser.id);

if (!_playerFrames[id].isNPC && _isBaseCombat && _isPlayerInvolved && !_isPlayersBase &&
id != CurrentUser.id)
{
var currentPercent:Number = _playerFrames[id].percent;
if (currentPercent > _firstBubbleThreshold && percent <= _firstBubbleThreshold)
showToast(ToastEnum.BUBBLE_ALERT, _bubblePrototype, _firstThresholdTitle,
_firstThresholdBody);
else if (currentPercent > _secondBubbleThreshold && percent <= _secondBubbleThreshold)
showToast(ToastEnum.BUBBLE_ALERT,
```

```
_bubblePrototype, _secondThresholdTitle, _secondThresholdBody);
else if (currentPercent > _thirdBubbleThreshold && percent <= _thirdBubbleThreshold)
showToast(ToastEnum.BUBBLE_ALERT, _bubblePrototype, _thirdThresholdTitle,
_thirdThresholdBody);
}


_playerFrames[id].percent = percent;
}
}

private function onBattleTimer( e:TimerEvent ):void
{
if (!presenter.battleRunning)
_battleTimer.stop();
else
{
_tempTime = presenter.battleTimeRemaining * .001 | 0;
_timeRemaining.text = (_tempTime / 60 | 0) + ":";
_tempTime = _tempTime % 60;
_timeRemaining.text = _timeRemaining.text + ((_tempTime > 9) ? _tempTime : "0" +
_tempTime);
}
}

private function onBattleStart():void
{
onBattleTimer(null);
_battleTimer.start();
}

private function onParticipantsAdded( id:String ):void
{
if (!_playerFrames.hasOwnProperty(id))
{
var player:PlayerVO = presenter.getPlayer(id);
addParticipantFrame(player, (_isPlayerInvolved) ? CurrentUser.faction : _overrideFaction);
layoutParticipants();
}
}

private function addParticipantFrame( player:PlayerVO, leftAttachFaction:String ):void
{
var sameFactionAsCurrentUser:Boolean = (player.faction == CurrentUser.faction);
if(_isInstancedMission)
sameFactionAsCurrentUser = (player.faction == leftAttachFaction);

var frame:PlayerBattleFrame = new PlayerBattleFrame(player, !sameFactionAsCurrentUser);

if (CurrentUser.id == player.id)
{
```

```
var fleet:FleetVO = presenter.getSelectedFleet();
frame.name = (fleet) ? fleet.name : player.name;
} else
frame.name = player.name;

frame.level = String(presenter.getParticipantRating(player.id));
frame.percent = presenter.getHealthPercentByPlayerID(player.id);
presenter.loadSmallImage(player.avatarName, frame.onAvatarLoaded);

if (presenter.isPlayerBaseOwner(player.id))
frame.setBubbleInformation(_firstBubbleThreshold, _secondBubbleThreshold,
_thirdBubbleThreshold);

_playerFrames[player.id] = frame;

if (leftAttachFaction == player.faction)
_leftAttach.addChild(frame);
else
_rightAttach.addChild(frame);
}

private function layoutParticipants():void
{
var i:uint;
var currentFrame:PlayerBattleFrame;
var yPos:Number;

if (_isPlayerInvolved)
yPos = 84;
else
yPos = 3;

for (; i < _leftAttach.numChildren; ++i)
{
currentFrame = PlayerBattleFrame(_leftAttach.getChildAt(i));
if (currentFrame.id == CurrentUser.id)
currentFrame.y = 3;
else
{
currentFrame.y = yPos;
yPos += currentFrame.height + 5;
}
}

yPos = 3;
for (i = 0; i < _rightAttach.numChildren; ++i)
{
currentFrame = PlayerBattleFrame(_rightAttach.getChildAt(i));
currentFrame.y = yPos;

yPos
```

```
+= currentFrame.height + 5;
}
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.TOP,
onStageResize));
}

private function onStageResize( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
}

private function sortByFaction( playerOne:PlayerVO, playerTwo:PlayerVO ):int
{
if (playerOne.id == CurrentUser.id)
return -1;

if (playerTwo.id == CurrentUser.id)
return -1;

if (playerOne.faction == CurrentUser.faction)
return -1;

if (playerTwo.faction == CurrentUser.faction)
return -1;

if (playerOne.faction != playerTwo.faction)
{
if (playerOne.faction == FactionEnum.SOVEREIGNTY)
return -1;
else if (playerTwo.faction == FactionEnum.SOVEREIGNTY)
return 1;

if (playerOne.faction == FactionEnum.TYRANNAR)
return -1;
else if (playerTwo.faction == FactionEnum.TYRANNAR)
return 1;

if (playerOne.faction == FactionEnum.IGA)
return -1;
else if (playerTwo.faction == FactionEnum.IGA)
return 1;
}

return 0;
}

override
```

```
public function destroy():void
{
presenter.removeListenerVitalPercentUpdates(onHealthUpdated);
presenter.removeListenerOnParticipantsAdded(onParticipantsAdded);
presenter.removeStartListener(onBattleStart);
super.destroy();

_battleTimer.stop();
_battleTimer = null;

_hitArea = null;

for (var id:String in _playerFrames)
{
_playerFrames[id].destroy();
_playerFrames[id] = null;
delete _playerFrames[id];
}
_playerFrames = null;

if (_timeRemainingHeader)
_timeRemainingHeader.destroy();

_timeRemainingHeader = null;

if (_timeRemaining)
_timeRemaining.destroy();

_timeRemaining = null;
}
}
}

-----------------------------------
File 734: igw\com\ui\hud\battle\BattleView.as
ï»¿package com.ui.hud.battle
{
import com.Application;
import com.enum.PositionEnum;
import com.ui.core.effects.EffectFactory;

import org.parade.util.DeviceMetrics;

public class BattleView extends BattleBaseView
{
private const MIN_X_POS:Number = 650;
private const EXIT_MIN_X_POS:Number = 67;

[PostConstruct]
override public function init():void
{
```

```
super.init();
x = DeviceMetrics.WIDTH_PIXELS * 0.5;

if (x < MIN_X_POS)
x = MIN_X_POS;

addHitArea();
addEffects();
effectsIN();
onResize();
}

private function onResize():void
{
this.scaleX = this.scaleY = Application.SCALE;
x = DeviceMetrics.WIDTH_PIXELS * 0.5;

if (x < MIN_X_POS)
x = MIN_X_POS;
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.CENTER, PositionEnum.TOP,
onResize));
}

override public function destroy():void
{
super.destroy();
//abilityView will handle destroying itself
}
}
}

-------------------------------------
File 735: igw\com\ui\hud\battle\PlayerBattleFrame.as
package com.ui.hud.battle
{
import com.enum.ui.PanelEnum;
import com.model.player.PlayerVO;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import
```

```
flash.display.BitmapData;
import flash.display.Sprite;
import flash.filters.ColorMatrixFilter;
import flash.geom.Point;
import flash.text.TextFormatAlign;

import org.shared.ObjectPool;

public class PlayerBattleFrame extends Sprite
{
private var _bg:ScaleBitmap;
private var _barBG:Bitmap;

private var _firstBubbleMarker:Bitmap;
private var _secondBubbleMarker:Bitmap;
private var _thirdBubbleMarker:Bitmap;

private var _avatarFrame:ScaleBitmap;

private var _avatar:ImageComponent;

private var _healthBar:ProgressBar;

private var _levelLbl:Label;
private var _nameLbl:Label;

private var _flipped:Boolean;

private var _player:PlayerVO;

private var _redFilter:ColorMatrixFilter;
private var _orangeFilter:ColorMatrixFilter;
private var _yellowFilter:ColorMatrixFilter;
private var _greenFilter:ColorMatrixFilter;

private var _currentFilter:ColorMatrixFilter;

private var _level:String = 'CodeString.Shared.Level'; //Level [[Number.Level]];

public function PlayerBattleFrame( player:PlayerVO, flip:Boolean = false )
{
super();

_player = player;

mouseEnabled = false;

_greenFilter = CommonFunctionUtil.getColorMatrixFilter(0x3cf219);
_yellowFilter = CommonFunctionUtil.getColorMatrixFilter(0xfbe81a);
_orangeFilter = CommonFunctionUtil.getColorMatrixFilter(0xfa7d0e);
_redFilter
```

```
= CommonFunctionUtil.getColorMatrixFilter(0xf81919);

_currentFilter = _greenFilter;

_flipped = flip;

_bg = UIFactory.getScaleBitmap((flip) ? PanelEnum.ENEMY_CONTAINER_NOTCHED :
PanelEnum.PLAYER_CONTAINER_NOTCHED);
_bg.width = 197;
_bg.height = 76;

_avatarFrame = UIFactory.getScaleBitmap(PanelEnum.CHARACTER_FRAME);
_avatarFrame.width = 75;
_avatarFrame.height = 75;

_avatar = ObjectPool.get(ImageComponent);
_avatar.init(70, 70);
_avatar.center = true;

_nameLbl = new Label(18, 0xffffff, 190, 33);
_nameLbl.bold = true;

_levelLbl = new Label(16, 0xf0f0f0, 190, 34);
_levelLbl.allCaps = true;

_barBG = UIFactory.getBitmap(PanelEnum.PLAYER_HEALTH_BG);
_barBG.width = 186;

var health:Bitmap = UIFactory.getBitmap(PanelEnum.STATBAR_GREY);
health.width = 176;
health.height = 21;

_healthBar = ObjectPool.get(ProgressBar);
_healthBar.init(ProgressBar.HORIZONTAL, health, null, 0.15, flip);
_healthBar.setMinMax(0, 1);
_healthBar.filters = [_currentFilter];

addChild(_bg);
addChild(_avatar);
addChild(_avatarFrame);
addChild(_nameLbl);
addChild(_levelLbl);
addChild(_barBG);
addChild(_healthBar);

layout();
}

private function layout():void
{
if
```

```
(!_flipped)
{
_bg.x = 78;

_levelLbl.align = TextFormatAlign.LEFT;

_nameLbl.align = TextFormatAlign.LEFT;

_barBG.x = _bg.x + 5;

_nameLbl.x = _levelLbl.x = _bg.x + 3;

_healthBar.x = _barBG.x + 5;
} else
{

_bg.x = 0;

_barBG.x = 5;
_avatar.x = _avatarFrame.x = _bg.x + _bg.width + 5;

_healthBar.x = _barBG.x + 5;

_levelLbl.align = TextFormatAlign.RIGHT;
_nameLbl.align = TextFormatAlign.RIGHT;

_levelLbl.x = _nameLbl.x = 2;
}

_levelLbl.y = 19;
}

public function onAvatarLoaded( asset:BitmapData ):void
{
if (_avatar && _avatarFrame && asset)
{
_avatar.onImageLoaded(asset);

_avatar.x = _avatarFrame.x + (_avatarFrame.width - _avatar.width) * 0.5;
_avatar.y = _avatarFrame.y + (_avatarFrame.height - _avatar.height) * 0.5;
}
}

public function set level( v:String ):void
{
if (_levelLbl)
{
_levelLbl.setTextWithTokens(_level, {'[[Number.Level]]':v});
if (_barBG)
_barBG.y = _levelLbl.y + _levelLbl.textHeight + 4;

if
```

```
(_healthBar)
_healthBar.y = _barBG.y + 4;
}
}
override public function set name( v:String ):void { _nameLbl.text = v; }
public function get percent():Number { return _healthBar.amount; }
public function set percent( v:Number ):void
{
_healthBar.amount = v;
var newFilter:ColorMatrixFilter = getHealthColor(v);
if (_currentFilter != newFilter)
{
_currentFilter = newFilter
_healthBar.filters = [_currentFilter];
}
}

public function setBubbleInformation( firstBubblePercent:Number,
secondBubblePercent:Number, thirdBubblePercent:Number ):void
{
_firstBubbleMarker = UIFactory.getBitmap('TabDivider');
_firstBubbleMarker.x = _healthBar.x + 176 * firstBubblePercent;
_firstBubbleMarker.y = _healthBar.y + (_healthBar.height - _firstBubbleMarker.height) * 0.5;

_secondBubbleMarker = UIFactory.getBitmap('TabDivider');
_secondBubbleMarker.x = _healthBar.x + 176 * secondBubblePercent;
_secondBubbleMarker.y = _healthBar.y + (_healthBar.height - _firstBubbleMarker.height) * 0.5;

_thirdBubbleMarker = UIFactory.getBitmap('TabDivider');
_thirdBubbleMarker.x = _healthBar.x + 176 * thirdBubblePercent;
_thirdBubbleMarker.y = _healthBar.y + (_healthBar.height - _firstBubbleMarker.height) * 0.5;

addChild(_firstBubbleMarker);
addChild(_secondBubbleMarker);
addChild(_thirdBubbleMarker);
}

private function getHealthColor( healthValue:Number ):ColorMatrixFilter
{

if (healthValue >= 0.75)
return _greenFilter;
else if (healthValue >= 0.50)
return _yellowFilter;
else if (healthValue >= 0.25)
return _orangeFilter;
else
return _redFilter;
}

public
```

```
function get isNPC():Boolean
{
return _player.isNPC;
}

public function get faction():String { return _player.faction; }
public function get id():String { return _player.id; }

public function destroy():void
{
_bg = null;
_barBG = null;
_firstBubbleMarker = null;
_secondBubbleMarker = null;
_thirdBubbleMarker = null;

_avatarFrame = null;

if (_avatar)
ObjectPool.give(_avatar);

_avatar = null;

if (_healthBar)
ObjectPool.give(_healthBar);

_healthBar = null;

if (_nameLbl)
_nameLbl.destroy();

_nameLbl = null;

if (_levelLbl)
_levelLbl.destroy();

_levelLbl = null;
}
}
}

-----------------------------------
File 736: igw\com\ui\hud\sector\SectorBaseView.as
package com.ui.hud.sector
{
import com.presenter.sector.ISectorPresenter;
import com.ui.core.View;

import org.parade.enum.ViewEnum;

public
```

```
class SectorBaseView extends View
{
[PostConstruct]
override public function init():void
{
super.init();
presenter.addCleanupListener(destroy);
}

[Inject]
public function set presenter( value:ISectorPresenter ):void { _presenter = value; }
public function get presenter():ISectorPresenter { return ISectorPresenter(_presenter); }

override public function get type():String { return ViewEnum.UI; }

override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
presenter.removeCleanupListener(destroy);
super.destroy();
}
}
}

-----------------------------------
File 737: igw\com\ui\hud\sector\SectorView.as
package com.ui.hud.sector
{
import com.Application;
import com.enum.CategoryEnum;
import com.enum.FactionEnum;
import com.enum.FleetStateEnum;
import com.enum.PositionEnum;
import com.enum.ToastEnum;
import com.enum.TypeEnum;
import com.enum.server.AllianceRankEnum;
import com.enum.server.AllianceResponseEnum;
import com.enum.ui.ButtonEnum;
import com.game.entity.components.battle.Attack;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Enemy;
import com.game.entity.components.shared.Move;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.game.entity.components.sector.Transgate;
import com.model.fleet.FleetVO;
import com.model.mission.MissionVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import
```

```
com.model.sector.SectorVO;
import com.model.starbase.BaseVO;
import com.model.prototype.IPrototype
import com.presenter.shared.IUIPresenter;
import com.presenter.starbase.IMissionPresenter;
import com.service.ExternalInterfaceAPI;
import com.service.language.Localization;
import com.service.server.incoming.data.SectorBattleData;
import com.ui.UIFactory;
import com.ui.alert.DropBubbleView;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.contextmenu.ContextMenu;
import com.ui.core.effects.EffectFactory;
import com.ui.modal.playerinfo.PlayerProfileView;
import com.util.CommonFunctionUtil;
import com.ui.core.ButtonPrototype;

import flash.display.Stage;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.ash.core.Entity;
import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class SectorView extends SectorBaseView
{
private var _battleServerAddress:String;
private var _joinBattleBtn:BitmapButton;
private var _stage:Stage;
private var _uiPresenter:IUIPresenter;
private var _missionPresenter:IMissionPresenter;

private var _tyrDestinations:Vector.<SectorVO>;
private var _sovDestinations:Vector.<SectorVO>;
private var _igaDestinations:Vector.<SectorVO>;

private var _privateDestinations:Vector.<SectorVO>;

private var _relocateSectorKey:String = "";
private var _relocateTransgateKey:String = "";
private var _sectorFaction:String = "";

private const MIN_X_POS:Number = 385;
private const MAX_BOOKMARKS:uint = 35;

private var _joinBattle:String = 'CodeString.FleetStatus.JoinBattleBtn'; //JOIN BATTLE

private
```

```
var _contextMenuStarbaseText:String = 'CodeString.ContextMenu.Sector.Starbase'; //Starbase
private var _contextMenuRecallText:String = 'CodeString.ContextMenu.Sector.Recall'; //Recall
private var _contextMenuEnterText:String = 'CodeString.ContextMenu.Sector.Enter'; //Enter
private var _contextMenuDefendText:String = 'CodeString.Alert.Battle.ViewBtn'; // Defend
private var _contextMenuJoinBattleText:String = 'CodeString.ContextMenu.Sector.JoinBattle';
//Join Battle
private var _contextMenuTackleText:String = "Scramble Warp"; //Join Battle
private var _contextMenuAttackText:String = 'CodeString.ContextMenu.Sector.Attack'; //Attack
private var _contextMenuWatchBattleText:String =
'CodeString.ContextMenu.Sector.WatchBattle'; //Watch Battle
private var _contextMenuTravelText:String = 'CodeString.ContextMenu.Sector.Travel'; //Travel
To:
private var _contextMenuFleetText:String = 'CodeString.ContextMenu.Sector.Fleet'; //Fleet

private var _contextMenuMissionSectorText:String =
'CodeString.ContextMenu.Sector.MissionSector'; //Mission Sector
private var _contextMenuDebrisText:String = 'CodeString.ContextMenu.Sector.CargoDebris';
//Cargo Debris
private var _contextMenuLoadCargoText:String = 'CodeString.ContextMenu.Sector.LoadCargo';
//Load Cargo
private var _contextMenuLinkCoords:String = 'CodeString.ContextMenu.Sector.LinkCoords'
//Link Coordinates

private var _sendMessage:String = 'CodeString.ContextMenu.Shared.SendMessage'; //Send
Message
private var _viewProfile:String = 'CodeString.ContextMenu.Chat.ViewProfile'; //View Profile
private var _addBookmark:String = 'CodeString.ContextMenu.Sector.AddBookmark'; //Add
Bookmark
private var _sectorNameText:String = "[[String.SectorName]] [[String.SectorEnum]]";
//[[String.SectorName]] [[String.SectorEnum]]

private var _toastAllianceRemoved:String = 'CodeString.Toast.AllianceRemoved'; //You have
been removed from your alliance.
private var _toastAllianceBadName:String = 'CodeString.Toast.AllianceBadName'; //Alliance
creation failed bad name.
private var _toastAllianceNameInUse:String = 'CodeString.Toast.AllianceNameInUse'; //Alliance
creation failed name in use.
private var _toastAllianceFailedOffline:String = 'CodeString.Toast.AllianceInviteFailedOffline';
//Alliance invite failed target offline.
private var _toastAllianceFailedIgnored:String = 'CodeString.Toast.AllianceInviteFailedIgnored';
//Alliance invite failed target is ignoring invites.
private var _toastAllianceAlreadyInAnAlliance:String =
'CodeString.Toast.AllianceInviteFailedAlreadyInAnAlliance'; //Alliance invite failed target is
already in an alliance.
private var _toastAllianceTooManyAlready:String =
'CodeString.Toast.AllianceJoinFailedTooManyAlready'; //Alliance join failed too many players in
alliance.
private var _toastAllianceNoLongerExists:String =
'CodeString.Toast.AllianceJoinFailedNoLongerExists'; //Alliance join failed it no longer exists
private var _toastAllianceInviteRecieved:String = 'CodeString.Toast.AllianceInviteRecieved';
//You
```

```
have been invited to join an alliance.
private var _toastAllianceJoined:String = 'CodeString.Toast.AllianceJoined'; //You have joined
an alliance.

private var _contextMenuRelocate:String = 'CodeString.ContextMenu.Sector.Relocate'
//Relocate
private var _acceptBtnText:String = 'CodeString.Shared.Accept'; //ACCEPT
private var _cancelBtnText:String = 'CodeString.Shared.CancelBtn'; //CANCEL
private var _relocateAlertTitle:String = 'CodeString.Alert.RelocateToTransgate.Title';
//RELOCATE
private var _relocateAlertBody:String = 'CodeString.Alert.RelocateToTransgate.Body'; //This will
move your starbase close to the selected player's starbase.\nAre you Sure?

private var _buyBtnText:String = 'CodeString.Shared.BuyPalladium'; //Buy Palladium
private var _notEnoughPalladiumTitle:String = 'CodeString.Alert.NotEnoughPalladium.Title';
//NOT ENOUGH PALLADIUM
private var _notEnoughPalladiumBody:String = 'CodeString.Alert.NotEnoughPalladium.Body';
//NOT ENOUGH PALLADIUM



[PostConstruct]
override public function init():void
{
super.init();
presenter.addBattleListener(onBattle);
presenter.addInteractListener(popContextMenuFromEntity);
presenter.addNotificationListener(onNotification);

_battleServerAddress = null;

_joinBattleBtn = UIFactory.getButton(ButtonEnum.RED_A, 640, 40, 0, 100, _joinBattle);
_joinBattleBtn.alpha = 0;
_joinBattleBtn.visible = false;
addChild(_joinBattleBtn);
addListener(_joinBattleBtn, MouseEvent.CLICK, onButtonClick);

{
var destinations:Vector.<SectorVO> = presenter.getTransgateDestinations();

_tyrDestinations = new Vector.<SectorVO>;
_sovDestinations = new Vector.<SectorVO>;
_igaDestinations = new Vector.<SectorVO>;

var len:uint = destinations.length;
var currentDestination:SectorVO;
for (var i:uint = 0; i < len; i++)
{
currentDestination = destinations[i];
if (currentDestination.id == presenter.sectorID)
{
```

```
_sectorFaction = currentDestination.sectorFaction;
continue;
}
switch (currentDestination.sectorFaction)
{
case FactionEnum.SOVEREIGNTY:
_sovDestinations.push(currentDestination);
break;
case FactionEnum.TYRANNAR:
_tyrDestinations.push(currentDestination);
break;
case FactionEnum.IGA:
_igaDestinations.push(currentDestination);
break;
}

}
}
{
var destinations:Vector.<SectorVO> = presenter.getPrivateDestinations();
_privateDestinations = new Vector.<SectorVO>;

var len:uint = destinations.length;
var currentDestination:SectorVO;
for (var i:uint = 0; i < len; i++)
{
currentDestination = destinations[i];
if (currentDestination.id == presenter.sectorID)
{
_sectorFaction = currentDestination.sectorFaction;
continue;
}
_privateDestinations.push(currentDestination);

}



}

_igaDestinations.sort(ordeSectors);
_tyrDestinations.sort(ordeSectors);
_sovDestinations.sort(ordeSectors);

_privateDestinations.sort(ordeSectors);

addHitArea();
presenter.addOnGenericAllianceMessageRecievedListener(onGenericAllianceMessage);
presenter.onBattle();
onResize();
addEffects();
```

```
effectsIN();
}


//=============================================================================
//*****************************************************************************
// ENTITY INTERACTION
//*****************************************************************************

//=============================================================================

public function popContextMenuFromEntity( x:int, y:int, entity:Entity, selectedEntity:Entity ):void
{
var detail:Detail = entity.get(Detail);
var entityPlayer:PlayerVO = presenter.getPlayer(detail.ownerID);

var selectedFleetHP:Number;
var selectedFleetVO:FleetVO;
var selectedFleetRating:int;
var selectedEntityLevel:int;
if (selectedEntity)
{

selectedFleetVO = presenter.getFleetVO(selectedEntity.id);
selectedFleetHP = selectedFleetVO.currentHealth;
selectedFleetRating = selectedFleetVO.level;

var selectedDetail:Detail = selectedEntity.get(Detail);
var selectedEntityPlayer:PlayerVO = presenter.getPlayer(selectedDetail.ownerID);
selectedEntityLevel = selectedEntityPlayer.level;
}
// don't allow commands when a fleet is being force recalled
var forceRecalling:Boolean = (selectedFleetVO && (selectedFleetVO.state ==
FleetStateEnum.FORCED_RECALLING));
var contextMenu:ContextMenu;
if (detail.category == CategoryEnum.SHIP)
{
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuFleetText, x, y, 150, _stage.stageWidth, _stage.stageHeight);
if (entity.has(Enemy))
{
if (Attack(entity.get(Attack)).inBattle)
{
contextMenu.addContextMenuChoice(_contextMenuWatchBattleText, presenter.watchBattle,
[entity]);

var entityAttack:Attack = Attack(entity.get(Attack));
if (canJoinBattle(entityAttack))
contextMenu.addContextMenuChoice(_contextMenuJoinBattleText, presenter.attackEntity,
[entity]);
```

```
} else if (selectedEntity && selectedFleetHP != 0 && !forceRecalling)
{
var isEnabled:Boolean = true;
//Make this work for tackling people he he
if (Move(entity.get(Move)) && Move(entity.get(Move)).moving)
contextMenu.addContextMenuChoice(_contextMenuTackleText, presenter.tackleEntity, [entity]);
/* else*/

if(entityPlayer && !entityPlayer.isNPC)
{
if(entityPlayer.level < 60 && entityPlayer.level * 3 < selectedEntityLevel * 2)
isEnabled = false;
else if(selectedEntityLevel < 60 && selectedEntityLevel * 3 < entityPlayer.level * 2)
isEnabled = false;
}

if (Move(entity.get(Move)) && !Move(entity.get(Move)).moving)
contextMenu.addContextMenuChoice(_contextMenuAttackText, presenter.attackEntity, [entity],
isEnabled);
}
} else if (entity.has(Owned))
{
if (forceRecalling)
{
contextMenu.destroy();
contextMenu = null;
} else if (!Attack(entity.get(Attack)).inBattle)
contextMenu.addContextMenuChoice(_contextMenuRecallText, presenter.recallFleet, [entity]);
else
contextMenu.addContextMenuChoice(_contextMenuJoinBattleText, presenter.watchBattle,
[entity]);
} else
{
if (Attack(entity.get(Attack)).inBattle)
contextMenu.addContextMenuChoice(_contextMenuWatchBattleText, presenter.watchBattle,
[entity]);
}
} else if (detail.waypointType != '')
{
switch (detail.waypointType)
{
case TypeEnum.WAYPOINT_TYPE_COLLECT:
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup("Mission", x, y, 184, _stage.stageWidth, _stage.stageHeight);
contextMenu.addContextMenuChoice("Collect", presenter.travelToWaypoint,
[presenter.selectedEntity, entity]);
break;
case TypeEnum.WAYPOINT_TYPE_ESCORT:
contextMenu
```

```
= ObjectPool.get(ContextMenu);
contextMenu.setup("Mission", x, y, 184, _stage.stageWidth, _stage.stageHeight);
contextMenu.addContextMenuChoice("Escort", presenter.travelToWaypoint,
[presenter.selectedEntity, entity]);
break;
case TypeEnum.WAYPOINT_TYPE_SCAN:
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup("Mission", x, y, 184, _stage.stageWidth, _stage.stageHeight);
contextMenu.addContextMenuChoice("Scan", presenter.travelToWaypoint,
[presenter.selectedEntity, entity]);
break;
}

} else
{
switch (detail.type)
{
case TypeEnum.STARBASE_SECTOR_IGA:
case TypeEnum.STARBASE_SECTOR_SOVEREIGNTY:
case TypeEnum.STARBASE_SECTOR_TYRANNAR:
if (entity.has(Owned))
{
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuStarbaseText, x, y, 150, _stage.stageWidth,
_stage.stageHeight);
if (selectedEntity && !Attack(entity.get(Attack)).inBattle)
{
contextMenu.addContextMenuChoice(_contextMenuRecallText, presenter.recallFleet,
[presenter.selectedEntity]);
if (entityPlayer && !entityPlayer.isNPC && selectedFleetVO && selectedFleetVO.defendTarget
!= entity.id)
contextMenu.addContextMenuChoice(_contextMenuDefendText, presenter.defendBase,
[presenter.selectedEntity, entity]);
}
if (!Attack(entity.get(Attack)).inBattle)
contextMenu.addContextMenuChoice(_contextMenuEnterText, presenter.enterStarbase,
[entity]);
else
contextMenu.addContextMenuChoice(_contextMenuJoinBattleText, presenter.watchBattle,
[entity]);
} else if (entity.has(Enemy))
{
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuStarbaseText, x, y, 150, _stage.stageWidth,
_stage.stageHeight);
if (selectedEntity && !Attack(entity.get(Attack)).bubbled)
{
var isEnabled:Boolean = true;
var baseAttackRatingDifferenceLimit:int =
presenter.getConstantPrototypeByName('baseAttackRatingDifferenceLimit');
var
```

```
baseAttackFreeForAllRatingThreshold:int =
presenter.getConstantPrototypeByName('baseAttackFreeForAllRatingThreshold');

var baseRating:uint = detail.baseRatingTech;
if(baseRating == 0)
{
baseRating = detail.baseLevel;
}
var diff:int = selectedFleetRating - baseRating;

if (entityPlayer && !entityPlayer.isNPC)
{
if (diff > baseAttackRatingDifferenceLimit && baseRating <
baseAttackFreeForAllRatingThreshold)
isEnabled = false;
}

if(entityPlayer && !entityPlayer.isNPC)
{
if(entityPlayer.level < 60 && entityPlayer.level * 3 < selectedEntityLevel * 2)
isEnabled = false;
else if(selectedEntityLevel < 60 && selectedEntityLevel * 3 < entityPlayer.level * 2)
isEnabled = false;
}

if(!Attack(entity.get(Attack)).inBattle)
{
contextMenu.addContextMenuChoice(_contextMenuAttackText, presenter.attackEntity, [entity],
isEnabled);
}
else if (entity.get(Attack).battle)
{
if(canJoinBattle(entity.get(Attack)))
{
contextMenu.addContextMenuChoice(_contextMenuJoinBattleText, presenter.attackEntity,
[entity], isEnabled);
}
}
}
} else if (Attack(entity.get(Attack)).inBattle)
{
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuStarbaseText, x, y, 150, _stage.stageWidth,
_stage.stageHeight);
contextMenu.addContextMenuChoice(_contextMenuWatchBattleText, presenter.watchBattle,
[entity]);
} else if (!entity.has(Enemy))
{
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuStarbaseText, x, y, 150, _stage.stageWidth,
_stage.stageHeight);
```

```
// defend anyone in your faction! what a nice guy you are
if (selectedEntity && !Attack(entity.get(Attack)).inBattle)
{
if (selectedFleetVO && selectedFleetVO.defendTarget != entity.id && !entityPlayer.isNPC)
contextMenu.addContextMenuChoice(_contextMenuDefendText, presenter.defendBase,
[presenter.selectedEntity, entity]);
}
}
break;
case TypeEnum.TRANSGATE_IGA:
case TypeEnum.TRANSGATE_SOVEREIGNTY:
case TypeEnum.TRANSGATE_TYRANNAR:

var transgateC:Transgate = entity.get(Transgate);

contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuTravelText, x, y, 184, _stage.stageWidth,
_stage.stageHeight);
if (selectedEntity)
{
var base:BaseVO = presenter.getBase(selectedFleetVO.starbaseID);
var mission:MissionVO = presenter.currentMission;

var _centerSpaceDestinations:Vector.<SectorVO>;
_centerSpaceDestinations = new Vector.<SectorVO>;
if (transgateC.customDestinationPrototypeGroup.length > 0)
{
var transgateCustomDestination:IPrototype =
presenter.getTransgateCustomDestinationPrototype(transgateC.customDestinationPrototypeGroup);
if(transgateCustomDestination)
{
var transgateCustomDestinationGroup:Vector.<IPrototype> =
presenter.getTransgateCustomDestinationGroupByCustomDestinationGroup(transgateCustomDestination
var customDestination:IPrototype;
for (var k:int = 0; k < transgateCustomDestinationGroup.length; k++)
{
customDestination = transgateCustomDestinationGroup[k];
var sectorKey:String = customDestination.getValue('sectorKey');

for (var j:uint = 0; j < _privateDestinations.length; j++)
{
if (_privateDestinations[j].id != sectorKey)
continue;

_centerSpaceDestinations.push(_privateDestinations[j]);
}

}
}
}
```

```
//ensure that we have something to add to the context menu
if (base.sectorID != presenter.sectorID || (_centerSpaceDestinations.length != 0 ||
_igaDestinations.length != 0 || _tyrDestinations.length != 0 || _sovDestinations.length != 0))
{
//see if this transgate is part of a mission
var len:uint;
var max:int;
var defaultIndex:int;
var currentDestination:SectorVO;
var loc:Localization = Localization.instance;
var i:uint = 0;
var current:int = Math.floor((Number(presenter.sectorID.split('.')[1]) - 1) / 3);

len = _centerSpaceDestinations.length;
if (len > 0)
{
if (_uiPresenter.csContextMenuDefaultIndex == -1)
{
if (current == 0)
defaultIndex = current;
else
{
max = len - 1;
if (current > max)
defaultIndex = max;
else
defaultIndex = current;
}
} else
{
defaultIndex = _uiPresenter.csContextMenuDefaultIndex;
max = len - 1;
if (defaultIndex > max)
defaultIndex = max;
}

contextMenu.addContextMenuMultiChoice(FactionEnum.IMPERIUM, defaultIndex,
_uiPresenter.setCSContextMenuDefaultIndex);
for (i = 0; i < len; i++)
{
currentDestination = _centerSpaceDestinations[i];
contextMenu.addChoiceToMultiChoice(FactionEnum.IMPERIUM,
loc.getString(currentDestination.sectorName) + " " + loc.
getString(currentDestination.sectorEnum),
presenter.travelViaTransgate, [currentDestination.id, presenter.selectedEntity, entity],
true, "", CommonFunctionUtil.getFactionColor(FactionEnum.IMPERIUM));
}

//contextMenu.addContextMenuChoice("Go 9001", presenter.travelViaTransgate, ["sector.9001",
presenter.selectedEntity,
```

```
entity]);
}


len = _igaDestinations.length;
if (len > 0)
{
if (_uiPresenter.igaContextMenuDefaultIndex == -1)
{
if (current == 0)
defaultIndex = current;
else
{
max = len - 1;
if (current > max)
defaultIndex = max;
else
defaultIndex = current;
}
} else
{
defaultIndex = _uiPresenter.igaContextMenuDefaultIndex;

max = len - 1;
if (defaultIndex > max)
defaultIndex = max;
}

contextMenu.addContextMenuMultiChoice(FactionEnum.IGA, defaultIndex,
_uiPresenter.setIGAContextMenuDefaultIndex);
for (i = 0; i < len; i++)
{
currentDestination = _igaDestinations[i];
contextMenu.addChoiceToMultiChoice(currentDestination.sectorFaction,
loc.getString(currentDestination.sectorName) + " " + loc.
getString(currentDestination.sectorEnum),
presenter.travelViaTransgate, [currentDestination.id, presenter.selectedEntity, entity],
true, "", CommonFunctionUtil.getFactionColor(currentDestination.sectorFaction));
}
}

len = _tyrDestinations.length;
if (len > 0)
{
if (_uiPresenter.tyrContextMenuDefaultIndex == -1)
{
if (current == 0)
defaultIndex = current;
else
{
max
```

```
= len - 1;
if (defaultIndex > max)
defaultIndex = max;
else
defaultIndex = current;
}
} else
{
defaultIndex = _uiPresenter.tyrContextMenuDefaultIndex;
max = len - 1;
if (current > max)
defaultIndex = max;
}

contextMenu.addContextMenuMultiChoice(FactionEnum.TYRANNAR, defaultIndex,
_uiPresenter.setTYRContextMenuDefaultIndex);
for (i = 0; i < len; i++)
{
currentDestination = _tyrDestinations[i];
contextMenu.addChoiceToMultiChoice(currentDestination.sectorFaction,
loc.getString(currentDestination.sectorName) + " " + loc.
getString(currentDestination.sectorEnum),
presenter.travelViaTransgate, [currentDestination.id, presenter.selectedEntity, entity],
true, "", CommonFunctionUtil.getFactionColor(currentDestination.sectorFaction));
}
}

len = _sovDestinations.length;
if (len > 0)
{
if (_uiPresenter.sovContextMenuDefaultIndex == -1)
{
if (current == 0)
defaultIndex = current;
else
{
max = len - 1;
if (current > max)
defaultIndex = max;
else
defaultIndex = current;
}
} else
{
defaultIndex = _uiPresenter.sovContextMenuDefaultIndex;
max = len - 1;
if (defaultIndex > max)
defaultIndex = max;
}

contextMenu.addContextMenuMultiChoice(FactionEnum.SOVEREIGNTY,
```

```
defaultIndex, _uiPresenter.setSOVContextMenuDefaultIndex);
for (i = 0; i < len; i++)
{
currentDestination = _sovDestinations[i];
contextMenu.addChoiceToMultiChoice(currentDestination.sectorFaction,
loc.getString(currentDestination.sectorName) + " " + loc.
getString(currentDestination.sectorEnum),
presenter.travelViaTransgate, [currentDestination.id, presenter.selectedEntity, entity],
true, "", CommonFunctionUtil.getFactionColor(currentDestination.sectorFaction));
}
}
_centerSpaceDestinations.length = 0;
_centerSpaceDestinations = null;


// show an option to hop to mission sector if it's not in your sector
if (mission && mission.accepted && !mission.rewardAccepted && mission.sector !=
presenter.sectorID)
contextMenu.addContextMenuChoice(_contextMenuMissionSectorText,
presenter.travelViaTransgate, [mission.sector, presenter.selectedEntity, entity]);

//show the option to recall the fleet if the fleet is now in its' home sector
if (base.sectorID != presenter.sectorID)
contextMenu.addContextMenuChoice(_contextMenuRecallText, presenter.recallFleet,
[selectedEntity, entity]);

//add check for same faction id with sector id
if(/*base.sectorID != presenter.sectorID && */CurrentUser.faction == _sectorFaction &&
presenter.neighborhood >= 0/*&& presenter.sectorID < 9000*/)
contextMenu.addContextMenuChoice(_contextMenuRelocate, relocateToTransgate,
[presenter.sectorID, entity]);
}
}
break;
case TypeEnum.TRADEDEPOT_IGA:
case TypeEnum.TRADEDEPOT_SOVEREIGNTY:
case TypeEnum.TRADEDEPOT_TYRANNAR:
/*contextMenu = ObjectPool.get(ContextMenu);
contextMenu.init('Trade Depot', x, y, _stage.stageWidth, _stage.stageHeight);
contextMenu.addContextMenuChoice('Coming Soon!', null, null);*/
break;
case TypeEnum.DERELICT_IGA:
case TypeEnum.DERELICT_SOVEREIGNTY:
case TypeEnum.DERELICT_TYRANNAR:
if (selectedEntity && selectedFleetHP != 0)
{
contextMenu = ObjectPool.get(ContextMenu);
contextMenu.setup(_contextMenuDebrisText, x, y, 150, _stage.stageWidth,
_stage.stageHeight);
contextMenu.addContextMenuChoice(_contextMenuLoadCargoText,
presenter.lootDerelictFleet,
```

```
[entity]);
}
break;
}
}

if (contextMenu)
{
contextMenu.addContextMenuChoice(_contextMenuLinkCoords, linkCoords, [entity]);
if (CurrentUser.bookmarkCount < MAX_BOOKMARKS)
contextMenu.addContextMenuChoice(_addBookmark, addBookmark, [entity]);

if (entityPlayer && !entityPlayer.isNPC && entityPlayer.id != CurrentUser.id)
contextMenu.addContextMenuChoice(_viewProfile, onViewProfile, [entityPlayer.id]);

_viewFactory.notify(contextMenu);
}

}


//==========================================================================================
//******************************************************************************************
// CONTROLS
//******************************************************************************************

//==========================================================================================

private function onButtonClick( e:MouseEvent ):void
{
switch (e.currentTarget)
{
case _joinBattleBtn:
presenter.joinBattle(_battleServerAddress);
_joinBattleBtn.enabled = false;
break;
}
}

private function onBattle( entity:Entity, added:Boolean, battleServerAddress:String ):void
{
if (added)
{
_battleServerAddress = battleServerAddress;
TweenLite.to(_joinBattleBtn, .2, {alpha:1.0, ease:Quad.easeIn});
} else
_joinBattleBtn.alpha = 0;

_joinBattleBtn.visible = added;
}

private
```

```
function onNotification( type:String, entity:Entity ):void
{
switch (type)
{
case 'bubble':
var view:DropBubbleView = DropBubbleView(showView(DropBubbleView, false));
view.entity = entity;
_viewFactory.notify(view);
break;
}
}
private function relocateToTransgate( sectorKey:String, entity:Entity):void
{
_relocateSectorKey = sectorKey;
_relocateTransgateKey = entity.id;

if( CurrentUser.wallet.premium < 100)
{
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_buyBtnText, openPalladiumShop, null, true,
ButtonEnum.GOLD_A));
buttons.push(new ButtonPrototype(_cancelBtnText));
showConfirmation(_notEnoughPalladiumTitle, _notEnoughPalladiumBody, buttons);
return;
}

var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_acceptBtnText, onRelocateToTransgate, null, true,
ButtonEnum.GOLD_A));
buttons.push(new ButtonPrototype(_cancelBtnText));
showConfirmation(_relocateAlertTitle, _relocateAlertBody, buttons);
}
private function openPalladiumShop():void
{
CommonFunctionUtil.popPaywall();
}
private function onRelocateToTransgate():void
{
if(_relocateTransgateKey.length > 0)
presenter.relocateToTransgate(_relocateSectorKey, _relocateTransgateKey);
}

private function linkCoords( entity:Entity ):void
{
if (entity)
{
var pos:Position = entity.get(Position);
if (pos && _uiPresenter)
_uiPresenter.linkCoords(int(pos.x * 0.01), int(pos.y * 0.01));

}
```

```
}

private function addBookmark( entity:Entity ):void
{
if (entity)
{
var pos:Position = entity.get(Position);
if (pos && _uiPresenter)
_uiPresenter.addBookmark(pos.x, pos.y);

}
}

private function onViewProfile( playerID:String ):void
{
var playerProfileView:PlayerProfileView =
PlayerProfileView(_viewFactory.createView(PlayerProfileView));
playerProfileView.playerKey = playerID;
_viewFactory.notify(playerProfileView);
}

private function onResize( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
var bounds:Rectangle = this.getBounds(this);
var pos:Number = MIN_X_POS * Application.SCALE;
x = (((DeviceMetrics.WIDTH_PIXELS - super.width) * .5 - bounds.x) > pos ?
((DeviceMetrics.WIDTH_PIXELS - super.width) * .5 - bounds.x) : pos);
}

private function ordeSectors( sectorOne:SectorVO, sectorTwo:SectorVO ):Number
{
if (!sectorOne)
return -1;
if (!sectorTwo)
return 1;

var sectorOneNumber:int = int(sectorOne.id.substr(7));
var sectorTwoNumber:int = int(sectorTwo.id.substr(7));

if (sectorOneNumber > sectorTwoNumber)
return 1;
else if (sectorOneNumber < sectorTwoNumber)
return -1;

return 0;
}

override protected function addEffects():void
{
```

```
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.CENTER, PositionEnum.TOP,
onResize));
}

private function onGenericAllianceMessage( messageEnum:int, allianceKey:String ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.SET_SUCCESS:
break;
case AllianceResponseEnum.KICKED:
case AllianceResponseEnum.LEFT:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceRemoved);
CurrentUser.alliance = '';
CurrentUser.allianceName = '';
CurrentUser.allianceRank = AllianceRankEnum.UNAFFILIATED;
CurrentUser.isAllianceOpen = false;
break;
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_BADNAME:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceBadName);
break;
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_NAMEINUSE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceNameInUse);
break;
case AllianceResponseEnum.INVITE_FAILED_OFFLINE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceFailedOffline);
break;
case AllianceResponseEnum.INVITE_FAILED_IGNORED:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceFailedIgnored);
break;
case AllianceResponseEnum.INVITE_FAILED_INALLIANCE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceAlreadyInAnAlliance);
break;
case AllianceResponseEnum.JOIN_FAILED_TOOMANYPLAYERS:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceTooManyAlready);
break;
case AllianceResponseEnum.JOIN_FAILED_NOALLIANCE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceNoLongerExists);
break;
case AllianceResponseEnum.INVITED:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceInviteRecieved);
break;
case AllianceResponseEnum.JOINED:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceJoined);
ExternalInterfaceAPI.shareAllianceJoin();
break;
}
}

private
```

```
function canJoinBattle( entityAttack:Attack ):Boolean
{

//TODO: Client guys, I don't know the client code well enough to do this, so this needs to only pop up in the following circumstances:
//1) My faction is part of the existing attack AND the number of FLEETS and BASES of that faction already involved is < maxPlayersPerFaction (Can't use players because one player may have both a fleet and a base, and that counts as two)
//2) My faction is NOT part of the existing attack AND the number of factions currently involved is < maxFactions
//Note that it IS allowed to join if the number of factions is > maxFactions, but my faction is one of the factions involved
if (entityAttack)
{
var currentBattle:SectorBattleData = entityAttack.battle;

if(currentBattle == null)
return false;

if(CurrentUser == null)
return false;

var entities:Array = currentBattle.participantFleets.concat(currentBattle.participantBase);
var canJoin:Boolean = true;
var currentPlayer:PlayerVO;
var i:uint;
var currentIGACount:int;
var currentSOVCount:int;
var currentTYRCount:int;
var factionCount:int;
var currentUsersFactionIsPresent:Boolean;

for (; i < entities.length; ++i)
{
if (entities[i] != null && entities[i] != '')
{
currentPlayer = presenter.getPlayer(Detail(presenter.getEntity(entities[i]).get(Detail)).ownerID);

if(currentPlayer != null)
{
switch (currentPlayer.faction)
{
case FactionEnum.IGA:
++currentIGACount;
break;
case FactionEnum.SOVEREIGNTY:
++currentSOVCount;
break;
case FactionEnum.TYRANNAR:
++currentTYRCount;
break;
```

```
}
}
}
}

if (currentIGACount > 0)
{
++factionCount;
if (CurrentUser.faction == FactionEnum.IGA)
{
currentUsersFactionIsPresent = true;
if (currentIGACount > currentBattle.maxPlayersPerFaction)
canJoin = false;
}
}

if (currentSOVCount > 0)
{
++factionCount;
if (CurrentUser.faction == FactionEnum.SOVEREIGNTY)
{
currentUsersFactionIsPresent = true;
if (currentSOVCount > currentBattle.maxPlayersPerFaction)
canJoin = false;
}
}
if (currentTYRCount > 0)
{
++factionCount;
if (CurrentUser.faction == FactionEnum.TYRANNAR)
{
currentUsersFactionIsPresent = true;
if (currentTYRCount > currentBattle.maxPlayersPerFaction)
canJoin = false;
}
}

if (!currentUsersFactionIsPresent && factionCount > currentBattle.maxFactions)
canJoin = false;

if (!currentBattle.joinable)
canJoin = false;

if (currentBattle.participantPlayers.length > currentBattle.maxPlayersPerFaction *
entityAttack.battle.maxFactions)
canJoin = false;

if (currentBattle.pvp == 0 && currentBattle.firstJoiningFactionPrototype.length > 0 &&
currentBattle.firstJoiningFactionPrototype != CurrentUser.faction)
canJoin
```

```
= false;

return canJoin;
} else
return false;
}

[Inject]
public function set stage( value:Stage ):void { _stage = value; }
[Inject]
public function set missionPresenter( v:IMissionPresenter ):void { _missionPresenter = v; }
[Inject]
public function set uiPresenter( v:IUIPresenter ):void { _uiPresenter = v; }

override public function destroy():void
{
presenter.removeOnGenericAllianceMessageRecievedListener(onGenericAllianceMessage);
super.destroy();

_tyrDestinations.length = 0;
_sovDestinations.length = 0;
_igaDestinations.length = 0;
_privateDestinations.length = 0;
_igaDestinations = _sovDestinations = _tyrDestinations = _privateDestinations = null;

_joinBattleBtn = UIFactory.destroyButton(_joinBattleBtn);
_missionPresenter = null;
_uiPresenter = null;
}
}
}

-----------------------------------
File 738: igw\com\ui\hud\sector\bookmarks\BookmarkEntry.as
package com.ui.hud.sector.bookmarks
{
import com.Application;
import com.controller.keyboard.KeyboardController;
import com.controller.keyboard.KeyboardKey;
import com.enum.ui.ButtonEnum;
import com.model.player.BookmarkVO;
import com.model.prototype.IPrototype;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import
```

```
flash.display.Bitmap;
import flash.display.Sprite;
import flash.display.Stage;
import flash.events.Event;
import flash.events.FocusEvent;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;

public class BookmarkEntry extends Sprite
{
public var gotoCoordsClicked:Signal;
public var changedBookmarkName:Signal;
public var fleetGotoCoords:Signal;
public var deleteBookmark:Signal;

private var _bg:Bitmap;

private var _name:Label;
private var _sectorName:Label;
private var _coords:Label;

private var _gotoCoordsBtn:BitmapButton;
private var _changeBookmarkName:BitmapButton;
private var _fleetGotoCoordsBtn:BitmapButton;
private var _deleteBookmark:BitmapButton;

private var _bookmark:BookmarkVO;

private var _stage:Stage;
private var _keyboard:KeyboardController;
private var _tooltip:Tooltips;

private var _sectorNameText:String = 'CodeString.Bookmarks.SectorName';
//[[SectorName:String]] [[SectorEnum:String]]
private var _sectorCoordsText:String = 'CodeString.Bookmarks.SectorCoords'; //[[SectorX:int]]
,[[SectorY:int]]

private var _gotoBookmarkText:String = 'CodeString.Bookmarks.GotoBookmarkTooltip'; //Goto
Bookmark
private var _selectAllText:String = 'CodeString.Bookmarks.SelectAllTooltip'; //Select All
Bookmark Text
private var _selectFleetText:String = 'CodeString.Bookmarks.SelectFleetTooltip'; //Selected
Fleet Goto Bookmark
private var _deleteBookmarkText:String = 'CodeString.Bookmarks.DeleteBookmarkTooltip';
//Delete Bookmark

public function BookmarkEntry( bookmark:BookmarkVO )
{
_bookmark
```

```
= bookmark;

gotoCoordsClicked = new Signal(BookmarkVO);
changedBookmarkName = new Signal(BookmarkVO);
fleetGotoCoords = new Signal(BookmarkVO);
deleteBookmark = new Signal(uint);

_bg = PanelFactory.getPanel('BookmarkRowBGBMD');

var sectorProto:IPrototype = _bookmark.sectorPrototype;
var faction:String = (sectorProto) ? sectorProto.getUnsafeValue('factionPrototype') : '';
var textColor:uint = (faction == '') ? 0xf0f0f0 : CommonFunctionUtil.getFactionColor(faction);

_name = new Label(18, textColor, 205, 26, true);
_name.align = TextFormatAlign.CENTER;
_name.x = 1;
_name.y = 1;
_name.maxChars = 20;
_name.restrict = "A-Za-z0-9'_\\- ";
_name.allowInput = true;
_name.clearOnFocusIn = false;
_name.addLabelColor(0xbdfefd, 0x000000);
_name.text = _bookmark.name;
_name.addEventListener(FocusEvent.FOCUS_OUT, onClearFocus, false, 0, true);

_sectorName = new Label(18, textColor, 108, 26, false);
_sectorName.align = TextFormatAlign.LEFT;
_sectorName.x = 209;
_sectorName.y = 1;
_sectorName.text = _bookmark.sectorName;
_sectorName.setTextWithTokens(_sectorNameText,
{'[[SectorName:String]]':_bookmark.sectorName,
'[[SectorEnum:String]]':_bookmark.sectorEnum});

_coords = new Label(18, textColor, 52, 26, true);
_coords.align = TextFormatAlign.LEFT;
_coords.x = 317;
_coords.y = 1;
_coords.setTextWithTokens(_sectorCoordsText, {'[[SectorX:int]]':_bookmark.displayX,
'[[SectorY:int]]':_bookmark.displayY});

_gotoCoordsBtn = ButtonFactory.getBitmapButton('GotoBtnUpBMD', 375, 4, '', 0,
'GotoBtnRollOverBMD', 'GotoBtnDownBMD', null, 'GotoBtnDownBMD');
_gotoCoordsBtn.addEventListener(MouseEvent.CLICK, onGotoCoordsClicked, false, 0, true);

_changeBookmarkName = ButtonFactory.getBitmapButton('EditBtnUpBMD', 417, 4, '', 0,
'EditBtnRollOverBMD', 'EditBtnUpBMD', null, 'EditBtnUpBMD');
_changeBookmarkName.addEventListener(MouseEvent.CLICK, onChangedBookmarkName,
false, 0, true);

_fleetGotoCoordsBtn
```

```
= UIFactory.getButton(ButtonEnum.FLEET_GOTO, 0, 0, 451, 4);
_fleetGotoCoordsBtn.addEventListener(MouseEvent.CLICK, onFleetGotoCoordsClicked, false,
0, true);

_deleteBookmark = ButtonFactory.getBitmapButton('DeleteBtnUpBMD', 498, 7, '', 0,
'DeleteBtnRollOverBMD', 'DeleteBtnDownBMD', null, 'DeleteBtnDownBMD');
_deleteBookmark.addEventListener(MouseEvent.CLICK, onDeleteClicked, false, 0, true);

addChild(_bg);
addChild(_name);
addChild(_sectorName);
addChild(_coords);
addChild(_gotoCoordsBtn);
addChild(_fleetGotoCoordsBtn)
addChild(_changeBookmarkName);
addChild(_deleteBookmark);
}

public function selectedFleet( v:Boolean ):void
{
_fleetGotoCoordsBtn.enabled = v;
}

private function onGotoCoordsClicked( e:MouseEvent ):void
{
gotoCoordsClicked.dispatch(_bookmark);
}

private function onFleetGotoCoordsClicked( e:MouseEvent ):void
{
fleetGotoCoords.dispatch(_bookmark);
}

private function onChangedBookmarkName( e:MouseEvent ):void
{
_stage.focus = _name;
_name.setSelection(0, _name.length);
}

private function onDeleteClicked( e:MouseEvent ):void
{
deleteBookmark.dispatch(_bookmark.index);
}

private function onClearFocus( e:FocusEvent = null ):void
{
setName();
}

public function set stage( value:Stage ):void { _stage = value; }
public
```

```
function get index():uint { return _bookmark.index; }

[Inject]
public function set tooltip( value:Tooltips ):void
{
_tooltip = value;

var loc:Localization = Localization.instance;
if (_gotoCoordsBtn)
_tooltip.addTooltip(_gotoCoordsBtn, this, null, loc.getString(_gotoBookmarkText));

if (_changeBookmarkName)
_tooltip.addTooltip(_changeBookmarkName, this, null, loc.getString(_selectAllText));

if (_fleetGotoCoordsBtn)
_tooltip.addTooltip(_fleetGotoCoordsBtn, this, null, loc.getString(_selectFleetText));

if (_deleteBookmark)
_tooltip.addTooltip(_deleteBookmark, this, null, loc.getString(_deleteBookmarkText));
}

[Inject]
public function set keyboard( value:KeyboardController ):void
{
_keyboard = value;
_keyboard.addKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);
}

private function onEnterPress( keyCode:uint ):void
{
if (_stage.focus == _name)
{
setName();
addEventListener(Event.ENTER_FRAME, removeFocus, false, 0, true);
}
}

private function setName():void
{
if (_bookmark.name != _name.text)
{
if (_name.text != '')
{
_bookmark.name = _name.text;
changedBookmarkName.dispatch(_bookmark);
} else
_name.text = _bookmark.name;
}
}

private
```

```
function removeFocus( e:Event ):void
{
removeEventListener(Event.ENTER_FRAME, removeFocus);
if (_stage)
_stage.focus = Application.STAGE;
}

public function destroy():void
{
if (gotoCoordsClicked)
gotoCoordsClicked.removeAll();

gotoCoordsClicked = null;

if (changedBookmarkName)
changedBookmarkName.removeAll();

changedBookmarkName = null;

if (deleteBookmark)
deleteBookmark.removeAll();

deleteBookmark = null;

if (_keyboard)
_keyboard.removeKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);

_keyboard = null;

if (_name)
{
_name.removeEventListener(FocusEvent.FOCUS_OUT, onClearFocus);
_name.destroy();
}

_name = null;

if (_sectorName)
_sectorName.destroy();

_sectorName = null;

if (_coords)
_coords.destroy();

_coords = null;

if (_gotoCoordsBtn)
_gotoCoordsBtn.destroy();

_gotoCoordsBtn
```

```
= null;

if (_changeBookmarkName)
_changeBookmarkName.destroy();

_changeBookmarkName = null;

if (_deleteBookmark)
_deleteBookmark.destroy();

_deleteBookmark = null;

_tooltip.removeTooltip(null, this);
_tooltip = null;
}
}
}

-----------------------------------
File 739: igw\com\ui\hud\sector\bookmarks\BookmarksView.as
package com.ui.hud.sector.bookmarks
{
import com.enum.ui.ButtonEnum;
import com.model.player.BookmarkVO;
import com.model.player.CurrentUser;
import com.presenter.shared.IBookmarkPresenter;
import com.ui.core.ButtonPrototype;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.display.Stage;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import org.shared.ObjectPool;

public class BookmarksView extends View
{
private var _bg:DefaultWindowBG;
private var _totalBookmarks:Label;

private var _bookmarkEntries:Vector.<BookmarkEntry>;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private
```

```
var _scrollRect:Rectangle;

private var _holder:Sprite;

private var _stage:Stage;

private var MAX_BOOKMARKS:uint = 35;

private var _titleText:String = 'CodeString.Bookmarks.Title'; //Bookmarks
private var _outOfString:String = 'CodeString.Shared.OutOf';
//[[Number.MinValue]]/[[Number.MaxValue]]

private var _totalBookmarkText:String = 'CodeString.Bookmarks.TotalBookmarks'; //Total
Bookmarks: [[Number.MinValue]]/[[Number.MaxValue]]
private var _deleteBookmarkNoBtnText:String = 'CodeString.Bookmarks.AddBookmark.NoBtn';
//NO
private var _deleteBookmarkRemoveBtnText:String =
'CodeString.Bookmarks.AddBookmark.Remove'; //REMOVE
private var _deleteBookmarkRemoveTitleText:String =
'CodeString.Bookmarks.AddBookmark.Title'; //DELETE BOOKMARK
private var _deleteBookmarkRemoveBodyText:String =
'CodeString.Bookmarks.AddBookmark.Body'; //Are you sure you want to remove this bookmark?

[PostConstruct]
override public function init():void
{
super.init();
_bookmarkEntries = new Vector.<BookmarkEntry>;

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(563, 245);
_bg.addTitle(_titleText, 114);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_totalBookmarks = new Label(16, 0xf0f0f0, 572, 25);
_totalBookmarks.align = TextFormatAlign.CENTER;
_totalBookmarks.y = 260;

_holder = new Sprite();
_holder.x = 25;
_holder.y = 53;
_maxHeight = 0;

_scrollRect = new Rectangle(_holder.x, _holder.y, 522, 197);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _bg.x + _bg.width - 28;
var
```

```
scrollbarYPos:Number = _bg.y + 52;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 7;

addChild(_bg);
addChild(_holder);
addChild(_scrollbar);
addChild(_totalBookmarks);

setUp(CurrentUser.bookmarks);

addEffects();
effectsIN();
}

private function setUp( v:Vector.<BookmarkVO> ):void
{
var len:uint = v.length;
var currentEntry:BookmarkEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = new BookmarkEntry(v[i]);
currentEntry.stage = _stage;
currentEntry.selectedFleet(presenter.hasSelectedFleet());
currentEntry.gotoCoordsClicked.add(onGotoCoordsClicked);
currentEntry.fleetGotoCoords.add(onFleetGotoCoordsClicked);
currentEntry.changedBookmarkName.add(onChangedBookmarkName);
currentEntry.deleteBookmark.add(onDeleteBookmark);
presenter.injectObject(currentEntry);
_holder.addChild(currentEntry);
_bookmarkEntries.push(currentEntry);
}
layout();
}

protected function layout():void
{
var len:uint = _bookmarkEntries.length;
var selection:BookmarkEntry;
var yPos:int = 0;
var xPos:int = _holder.x;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _bookmarkEntries[i];
selection.x = xPos;
selection.y
```

```
= yPos;
_maxHeight += selection.height - 1;
yPos += selection.height - 1;
}
_maxHeight += 1
_scrollbar.updateScrollableHeight(_maxHeight);
_totalBookmarks.setTextWithTokens(_totalBookmarkText, {'[[Number.MinValue]]':len,
'[[Number.MaxValue]]':MAX_BOOKMARKS});

if (_maxHeight <= _scrollRect.height)
_scrollbar.resetScroll();
else
onChangedScroll(_scrollbar.percent);

}

private function onGotoCoordsClicked( v:BookmarkVO ):void
{
presenter.gotoCoords(v.x, v.y, v.sector);
destroy();
}

private function onFleetGotoCoordsClicked( v:BookmarkVO ):void
{
presenter.fleetGotoCoords(v.x, v.y, v.sector);
destroy();
}

private function onChangedBookmarkName( v:BookmarkVO ):void
{
presenter.updateBookmark(v);
}

private function onDeleteBookmark( v:uint ):void
{
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_deleteBookmarkRemoveBtnText, deleteBookmark, [v], true,
ButtonEnum.GREEN_A));
buttons.push(new ButtonPrototype(_deleteBookmarkNoBtnText));
showConfirmation(_deleteBookmarkRemoveTitleText, _deleteBookmarkRemoveBodyText,
buttons);
}

private function deleteBookmark( v:uint ):void
{
var len:uint = _bookmarkEntries.length;
var selection:BookmarkEntry;
for (var i:uint = 0; i < len; ++i)
{
selection = _bookmarkEntries[i];
if
```

```
(selection.index == v)
{
_holder.removeChild(selection);
_bookmarkEntries.splice(i, 1);
selection.destroy();
selection = null;
presenter.deleteBookmark(v);
layout();
break;
}
}
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IBookmarkPresenter ):void { _presenter = v; }
public function get presenter():IBookmarkPresenter { return IBookmarkPresenter(_presenter); }

[Inject]
public function set stage( value:Stage ):void { _stage = value; }

override public function destroy():void
{
super.destroy();

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_totalBookmarks)
_totalBookmarks.destroy();

_totalBookmarks = null;

var len:uint = _bookmarkEntries.length;
var currentEntry:BookmarkEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _bookmarkEntries[i];
currentEntry.destroy();
currentEntry = null;
}
```

```
_bookmarkEntries.length = 0;

_scrollbar = null;
_maxHeight = 0;

_scrollRect = null;

_holder = null;
}
}
}

------------------------------------
File 740: igw\com\ui\hud\shared\ChatView.as
package com.ui.hud.shared
{
import com.Application;
import com.controller.keyboard.KeyboardKey;
import com.enum.PositionEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.StateEvent;
import com.model.chat.ChatChannelVO;
import com.model.chat.ChatPanelVO;
import com.model.motd.MotDVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.presenter.shared.IChatPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.contextmenu.ContextMenu;
import com.ui.core.component.label.Label;
import com.ui.core.component.tab.TabComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.core.effects.EffectFactory;
import com.ui.modal.alliances.alliance.AllianceView;
import com.ui.modal.alliances.noalliance.NoAllianceView;
import com.ui.modal.ignore.IgnoreListView;
import com.ui.modal.information.MessageOfTheDayView;
import com.ui.modal.playerinfo.PlayerProfileView;

import flash.display.Bitmap;
import flash.display.BlendMode;
import flash.display.DisplayObject;
import
```

```
flash.display.Sprite;
import flash.display.Stage;
import flash.events.FocusEvent;
import flash.events.MouseEvent;
import flash.events.TextEvent;
import flash.geom.Point;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;
import org.greensock.TweenLite;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

import com.enum.server.AllianceRankEnum;
import com.enum.ToastEnum;

public class ChatView extends View
{

public static const SECTOR_TAB:String = "SectorTab";
public static const ALLIANCE_TAB:String = "AllianceTab";

public static const HALF:Number = 0;
public static const FULL:Number = 1;
public static const MIN:Number = 2;

private var _tabs:TabComponent;

private var _chatLog:Label;
private var _textInput:Label;
private var _motd:Label;

private var _fullBtn:BitmapButton;
private var _halfBtn:BitmapButton;
private var _minBtn:BitmapButton;
private var _ignorePlayerListBtn:BitmapButton;
private var _unselectedTab:BitmapButton;
private var _alertedBtn:BitmapButton;

private var _stage:Stage;

private var _inputBG:ScaleBitmap;
private var _chatBG:ScaleBitmap;
private var _motdBG:ScaleBitmap;

private var _textAlertColor:uint;

private
```

```
var _motdHitArea:Sprite;
private var _motdHolder:Sprite;
private var _chatHolder:Sprite;

private var _scrollbar:VScrollbar;

private var _scrollRect:Rectangle;

private var _minYPos:Number;

private var _currentChatWindowState:uint;
private var _selectedTab:int;

private var _chatPanels:Vector.<ChatPanelVO>;
private var _motdMessages:Vector.<MotDVO>;
private var _animating:Boolean;
private var _currentMOTDIndex:int;
private var shownMOTD:Boolean;

private var _tooltip:Tooltips;

private const MIN_Y_POS:Number = 467;

private var _enterMessage:String = 'CodeString.Comms.EnterMessage'; //Enter Message
private var _block:String = 'CodeString.ContextMenu.Chat.Block'; //Block
private var _unblock:String = 'CodeString.ContextMenu.Chat.Unblock'; //Unblock
private var _muteString:String = 'CodeString.ContextMenu.Chat.Mute'; //Mute
private var _unmute:String = 'CodeString.ContextMenu.Chat.Unmute'; //Unmute
private var _viewProfile:String = 'CodeString.ContextMenu.Chat.ViewProfile'; //View Profile
private var _sendMessage:String = 'CodeString.ContextMenu.Shared.SendMessage'; //Send
Message

private var _ignoreListTooltip:String = 'CodeString.Chat.IgnoreListTooltip'; //Ignored Player List

private var _tooLowAllianceRankText:String = 'CodeString.Toast.TooLowAllianceRank'; //Too
Low Alliance Rank. You need to be promoted first

[PostConstruct]
override public function init():void
{
super.init();
_motdHitArea = new Sprite();
_motdHitArea.graphics.beginFill(0xf0f0f0, 0.0);
_motdHitArea.graphics.drawRect(0, 0, 461, 30);
_motdHitArea.graphics.endFill();
_motdHitArea.x = 9;
_motdHitArea.y = -24;
addListener(_motdHitArea, MouseEvent.CLICK, showMotDView);
addListener(_motdHitArea, MouseEvent.ROLL_OVER, onMOTDRollOver);
addListener(_motdHitArea, MouseEvent.ROLL_OUT, onMOTDRollOut);

_motdHolder
```

```
= new Sprite();
_motdHolder.mouseEnabled = false;
_motdHolder.x = _motdHitArea.x;
_motdHolder.y = _motdHitArea.y;

_motd = new Label(18, 0xfef9bd, 453, 30);
_motd.alpha = 0;
_motd.constrictTextToSize = false;
_motd.align = TextFormatAlign.LEFT;
_motd.letterSpacing = 1.5;
_motdHolder.addChild(_motd);

_tabs = ObjectPool.get(TabComponent);
_tabs.init(PanelEnum.CONTAINER_RIGHT_NOTCHED_ARROW,
PanelEnum.HEADER_NOTCHED, 515, 156, 32);

_chatPanels = presenter.chatPanels;
var len:uint = _chatPanels.length;
var xPos:Number;
var currentPanel:ChatPanelVO;
var headerToUse:String;
for (var i:uint = 0; i < len; ++i)
{
currentPanel = _chatPanels[i];
if (i == 0)
{
headerToUse = ButtonEnum.HEADER_NOTCHED;
_tabs.addTab(String(currentPanel.id), headerToUse, 60, 32, xPos, 0, currentPanel.name,
LabelEnum.H2);
}
else
{
xPos = 61 + (i-1) * 101;
headerToUse = ButtonEnum.HEADER;
_tabs.addTab(String(currentPanel.id), headerToUse, 100, 32, xPos, 0, currentPanel.name,
LabelEnum.H2);
}
}
_tabs.addSwitchTabListener(onTabSwitched);

_fullBtn = UIFactory.getButton(ButtonEnum.ICON_WINDOW_HALF, 0, 0, 496, 13);
_halfBtn = UIFactory.getButton(ButtonEnum.ICON_WINDOW_FULL, 0, 0, 496, 13);
_minBtn = UIFactory.getButton(ButtonEnum.ICON_WINDOW_MIN, 0, 0, 481, 13);
_ignorePlayerListBtn = UIFactory.getButton(ButtonEnum.ICON_IGNORE, 0, 0, 5, _tabs.height -
35);
_halfBtn.visible = false;

_tooltip.addTooltip(_ignorePlayerListBtn, this, null,
Localization.instance.getString(_ignoreListTooltip));

addListener(_fullBtn,
```

```
MouseEvent.CLICK, onMouseClick);
addListener(_halfBtn, MouseEvent.CLICK, onMouseClick);
addListener(_minBtn, MouseEvent.CLICK, onMouseClick);
addListener(_ignorePlayerListBtn, MouseEvent.CLICK, popChatContextMenu);

_inputBG = UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED, 474, 30);
_inputBG.x = 35;
_inputBG.y = _tabs.height - 36;

_chatHolder = new Sprite();
_chatHolder.x = 9;
_chatHolder.y = 45;

_chatBG = UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL, 487,
107);
_chatBG.x = 6;
_chatBG.y = 41;

_chatLog = new Label(12, 0xf0f0f0, 480, 95, false, 1);
_chatLog.autoSize = TextFieldAutoSize.LEFT
_chatLog.align = TextFormatAlign.LEFT;
_chatLog.selectable = true;
_chatLog.mouseEnabled = true;
_chatLog.constrictTextToSize = false;
_chatLog.multiline = true;
_chatLog.addEventListener(TextEvent.LINK, onHyperLinkEvent, false, 0, true);
_chatHolder.addChild(_chatLog);
_chatLog.y -= 2;

_scrollRect = new Rectangle(0, 0, 490, 95);
_scrollRect.y = 0;
_chatHolder.scrollRect = _scrollRect

_textInput = new Label(13, 0xbdfefd, 470, 21, true, 1);
_textInput.text = _enterMessage;
_textInput.allowInput = true;
_textInput.clearOnFocusIn = true;
_textInput.letterSpacing = .8;
_textInput.addLabelColor(0xbdfefd, 0x000000);
_textInput.x = 40;
_textInput.y = _tabs.height - 33;
addListener(_textInput, FocusEvent.FOCUS_IN, onInputGainedFocus);
addListener(_textInput, FocusEvent.FOCUS_OUT, onInputLostFocus);

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(5, 5, 2, 2);
var scrollbarXPos:Number = _tabs.width - 21;
var scrollbarYPos:Number = _chatBG.y;
_scrollbar.init(7, _chatBG.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', true, this, _chatHolder);
_scrollbar.onScrollSignal.add(onChangedScroll);
```

```
_scrollbar.updateDisplayedHeight(_chatBG.height);
_scrollbar.updateScrollableHeight(_chatLog.textHeight);
_scrollbar.maxScroll = 16;

_keyboard.addKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);

addChild(_tabs);
addChild(_fullBtn);
addChild(_halfBtn);
addChild(_minBtn);
addChild(_ignorePlayerListBtn);
addChild(_inputBG);
addChild(_motdHitArea);
addChild(_motdHolder);
addChild(_textInput);
addChild(_scrollbar);
addChild(_chatBG);
addChild(_chatHolder);

_motdMessages = presenter.motdMessages;
if (_motdMessages && _motdMessages.length > 0)
{
_motdHitArea.buttonMode = true;
startMOTDFade();
}

presenter.addMotDUpdatedListener(onNewMOTDMessageAdded);
presenter.addChatListener(onChatUpdate);
presenter.addOnDefaultChannelUpdatedListener(onDefaultChannelUpdated);
_tabs.setSelectedTab('0');

addEffects();
effectsIN();
onResize();

visible = !presenter.inFTE;
}


//==========================================================================
//**************************************************************************************************
// GENERAL
//**************************************************************************************************

//==========================================================================

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.BOTTOM,
onResize));
```

```
}

private function onTabSwitched( name:String ):void
{
var panelIndex:int = int(name);
if (_selectedTab != panelIndex)
{

if ((panelIndex == 3 || panelIndex == 4) && CurrentUser.alliance == '')
{
_tabs.setSelectedTab('0');
var noAllianceView:NoAllianceView =
NoAllianceView(_viewFactory.createView(NoAllianceView));
_viewFactory.notify(noAllianceView);
}
else if (panelIndex == 4 && CurrentUser.allianceRank < AllianceRankEnum.MEMBER)
{
_tabs.setSelectedTab('3');
showToast(ToastEnum.WRONG, null, _tooLowAllianceRankText);
} else
{
_selectedTab = int(name);
var selectedTab:ChatPanelVO;
if (_selectedTab < _chatPanels.length)
{
selectedTab = _chatPanels[_selectedTab];
presenter.defaultChannel = selectedTab.defaultSendChannel;
onChatUpdate(_selectedTab, presenter.getPanelLogs(_selectedTab), false);
}

if (_alertedBtn)
TweenLite.killTweensOf(_alertedBtn);

}
}

}

private function selectChannel( selectedChannelArgs:Array ):void
{
var currentChannel:ChatChannelVO = selectedChannelArgs[0];
if (currentChannel != null)
{
var color:uint = selectedChannelArgs[0].channelColor;
_textInput.updateLabelColor(color);
_textInput.updateLabelSelectionColor(color);
presenter.defaultChannel = currentChannel;
}
}

private
```

```
function onDefaultChannelUpdated( v:ChatChannelVO ):void
{
var color:uint = v.channelColor;
_textInput.updateInputText(v.displayName);
_textInput.updateLabelColor(color);
_textInput.updateLabelSelectionColor(color);
}

private function onEnterPress( keyCode:uint ):void
{
if (!presenter.hudEnabled)
return;
if (_stage.focus == null || _stage.focus == _textInput || _stage.focus == Application.STAGE)
{
if (_stage.focus != _textInput)
{
_stage.focus = _textInput;
_textInput.setSelection(_textInput.length, _textInput.length);
} else
{
if (_textInput.text != '')
{
presenter.sendChatMessage(StringUtil.escapeHTML(_textInput.text));
_textInput.text = '';
}
}
}
}

private function onChatUpdate( panelID:int, chat:String, update:Boolean ):void
{
if (_selectedTab == panelID)
{
_chatLog.htmlText = chat;
_scrollbar.updateScrollableHeight(_chatLog.textHeight);
if (_scrollbar.percent == 1)
{
onChangedScroll(_scrollbar.percent);
}
} else
{
_alertedBtn = _tabs.getTab(panelID.toString());
_textAlertColor = _alertedBtn.textColor;
onFadeOut();
}
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_chatLog.textHeight - _scrollRect.height) * percent;
_chatHolder.scrollRect
```

```
= _scrollRect;
}

private function onResize():void
{
this.scaleX = this.scaleY = Application.SCALE;
var yPos:Number;
switch (_currentChatWindowState)
{
case HALF:
case FULL:
yPos = DeviceMetrics.HEIGHT_PIXELS - (_tabs.height * Application.SCALE);
break;
case MIN:
yPos = DeviceMetrics.HEIGHT_PIXELS - (33 * Application.SCALE);
break;
}

y = (yPos < MIN_Y_POS) ? MIN_Y_POS : yPos;
}

private function popChatContextMenu( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
_viewFactory.notify(IgnoreListView(_viewFactory.createView(IgnoreListView)));
e.stopPropagation();
}

private function onInputGainedFocus( e:FocusEvent ):void { presenter.chatHasFocus = true; }
private function onInputLostFocus( e:FocusEvent ):void { presenter.chatHasFocus = false; }


//=========================================================================================
//*****************************************************************************************
// CHAT WINDOW MANIPULATION
//*****************************************************************************************

//=========================================================================================

private function onMouseClick( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
switch (e.target)
{
case _fullBtn:
updateWindowState(FULL);
TweenLite.to(_tabs, .2, {height:249, onUpdate:onMoveTweenUpdate, overwrite:0});
TweenLite.to(_chatBG, .2, {height:200, overwrite:0});
TweenLite.to(_scrollRect,
```

```
.2, {height:178, onUpdate:onScrollRectUpdate, onComplete:onTweenComplete, overwrite:0});
break;
case _halfBtn:
updateWindowState(HALF);
TweenLite.to(_tabs, .2, {height:156, onUpdate:onMoveTweenUpdate, overwrite:0});
TweenLite.to(_chatBG, .2, {height:107, overwrite:0});
TweenLite.to(_scrollRect, .2, {height:90, onUpdate:onScrollRectUpdate,
onComplete:onTweenComplete, overwrite:0});
break;
case _minBtn:
updateWindowState(MIN);
TweenLite.to(_tabs, .2, {height:24, onUpdate:onMoveTweenUpdate, overwrite:0});
TweenLite.to(_chatBG, .2, {height:1, overwrite:0});
TweenLite.to(_scrollRect, .2, {height:1, onUpdate:onScrollRectUpdate,
onComplete:onTweenComplete, overwrite:0});
break;
}
e.stopPropagation();
}

private function onFadeOut():void
{
if (_alertedBtn)
{
_alertedBtn.textColor = 0xfff2dd;
TweenLite.to(_alertedBtn, 0.5, {onComplete:onFadeIn});
}
}

private function onFadeIn():void
{
if (_alertedBtn)
{
_alertedBtn.textColor = _textAlertColor;
TweenLite.to(_alertedBtn, 0.5, {onComplete:onFadeOut});
}
}

private function onMoveTweenUpdate():void
{
var yPos:Number;
if (_currentChatWindowState == MIN)
{
var displayObject:DisplayObject = _tabs.panelBG;
yPos = DeviceMetrics.HEIGHT_PIXELS - (displayObject.height * Application.SCALE);
y = (yPos < MIN_Y_POS) ? MIN_Y_POS : yPos;
} else
{
_inputBG.y = _tabs.height - 36;
_textInput.y = _tabs.height - 33;
_ignorePlayerListBtn.y
```

```
= _tabs.height - 35;
yPos = DeviceMetrics.HEIGHT_PIXELS - (_tabs.height * Application.SCALE);
y = (yPos < MIN_Y_POS) ? MIN_Y_POS : yPos;
}

}

private function onScrollRectUpdate():void
{
_chatHolder.scrollRect = _scrollRect;
}

private function updateWindowState( newState:uint ):void
{
_currentChatWindowState = newState;

_scrollbar.visible = false;
switch (_currentChatWindowState)
{
case HALF:
_halfBtn.visible = false;

_minBtn.x = 481;
_minBtn.visible = true;

_fullBtn.x = 496;
_fullBtn.visible = true;
break;
case FULL:
_fullBtn.visible = false;

_minBtn.x = 481;
_minBtn.visible = true;

_halfBtn.x = 496;
_halfBtn.visible = true;
break;
case MIN:
_minBtn.visible = false;

_halfBtn.x = 481;
_halfBtn.visible = true;

_fullBtn.x = 496;
_fullBtn.visible = true;
break;
}
TweenLite.killTweensOf(_tabs);
}

private
```

```
function onScrollUp( e:MouseEvent ):void
{
_scrollbar.scrollUp();
}

private function onScrollDown( e:MouseEvent ):void
{
_scrollbar.scrollDown();
}

private function onTweenComplete():void
{
_chatHolder.y = 55;
_scrollRect.y = (_chatLog.textHeight - _scrollRect.height) * _scrollbar.percent;
_chatHolder.scrollRect = _scrollRect;

_scrollbar.updateScrollbarHeight(_chatBG.height);
_scrollbar.updateScrollableHeight(_chatLog.textHeight);
_scrollbar.updateDisplayedHeight(_chatBG.height);

_scrollbar.visible = true;

if (_currentChatWindowState == MIN)
{
var newYPos:Number = DeviceMetrics.HEIGHT_PIXELS - (33 * Application.SCALE);
y = (newYPos < MIN_Y_POS) ? MIN_Y_POS : newYPos;
}
}

override protected function onStateChange( state:String ):void
{
if (state == StateEvent.GAME_BATTLE)
{
TweenLite.killTweensOf(_motd);
_motd.visible = false;
} else
{
_motd.visible = true;
startMOTDFade();
}
}


//==========================================================================
//**************************************************************************************************
// CHAT HYPERLINKS
//**************************************************************************************************

//==========================================================================

private
```

```
function onHyperLinkEvent( e:TextEvent ):void
{
e.stopPropagation();
var indexToSplitAt:int = e.text.indexOf('.');
if (indexToSplitAt != -1 && e.text.length > (indexToSplitAt + 1))
{
var linkType:String = e.text.slice(0, indexToSplitAt);
var text:String = e.text.slice(indexToSplitAt + 1);
if(linkType == "NameLink")
{
if (text.indexOf(':') != -1)
{
var nameLinkArgs:Array = text.split(':');
var playerName:String = nameLinkArgs[1];
var playerKey:String = nameLinkArgs[0];

if (CurrentUser.id != playerKey)
{
var currentPlayer:PlayerVO = presenter.getPlayer(playerKey);
var target:* = e.target;
var parent:* = e.target.parent;

var location:Point = new Point(parent.mouseX, parent.mouseY);
location = parent.localToGlobal(location);

var nameContext:ContextMenu = ObjectPool.get(ContextMenu);
nameContext.setup(playerName, location.x, location.y, 150, DeviceMetrics.WIDTH_PIXELS,
DeviceMetrics.HEIGHT_PIXELS);
if (presenter.isBlocked(playerKey))
{
nameContext.addContextMenuChoice(_unblock, onBlockPlayer, [playerKey]);
} else
{
nameContext.addContextMenuChoice(_block, onBlockPlayer, [playerKey]);
}


if (presenter.isMuted(playerKey))
{
nameContext.addContextMenuChoice(_unmute, onMutePlayer, [playerKey]);
} else
{
nameContext.addContextMenuChoice(_muteString, onMutePlayer, [playerKey]);
}
nameContext.addContextMenuChoice(_viewProfile, onViewProfile, [playerKey]);
_viewFactory.notify(nameContext);
} else
{
onViewProfile(CurrentUser.id);
}
}
```

```
}
else if (linkType == "AllianceLink")
{
onViewAlliance("alliance." + text);
}
else if (linkType == "CoordLink")
{
if (!presenter.inFTE)
{
var coords:Array = text.split(',');
presenter.gotoCoords(int(coords[0] * 100), int(coords[1] * 100), coords[2]);
}
}
}
}

private function onBlockPlayer( playerID:String ):void
{
presenter.blockOrUnblockPlayer(playerID);
}

private function onMutePlayer( playerID:String ):void
{
presenter.mutePlayer(playerID);
}

private function onViewProfile( playerID:String ):void
{
var playerProfileView:PlayerProfileView =
PlayerProfileView(_viewFactory.createView(PlayerProfileView));
playerProfileView.playerKey = playerID;
_viewFactory.notify(playerProfileView);
}

private function onViewAlliance( allianceKey:String ):void
{
var allianceView:AllianceView = AllianceView(_viewFactory.createView(AllianceView));
allianceView.allianceKey = allianceKey;
_viewFactory.notify(allianceView);
}


//==========================================================================
//**************************************************************************************************
// MOTD
//**************************************************************************************************

//==========================================================================

private
```

```
function onMOTDRollOut( e:MouseEvent ):void
{
_motd.textColor = 0xfef9bd;
}

private function onMOTDRollOver( e:MouseEvent ):void
{
_motd.textColor = 0xffffe6;
}

private function showMotDView( e:MouseEvent = null ):void
{
if (_motdMessages && _motdMessages.length > 0)
{
var view:MessageOfTheDayView =
MessageOfTheDayView(_viewFactory.createView(MessageOfTheDayView));
_viewFactory.notify(view);
}
e.stopPropagation();
}

private function onNewMOTDMessageAdded( motd:Vector.<MotDVO> ):void
{
_motdMessages = presenter.motdMessages;

if (_animating)
_currentMOTDIndex = -1;
else
{
_motdHitArea.buttonMode = true;
startMOTDFade();
}

/*if (!presenter.inFTE && !StarbaseCommand.shownMOTD)
{
presenter.dispatch(new StarbaseEvent(StarbaseEvent.WELCOME_BACK));
}*/
}

private function startMOTDFade():void
{
if (_motdMessages && _motdMessages.length > 0)
{
_animating = true;
_currentMOTDIndex = 0;
_motd.text = _motdMessages[_currentMOTDIndex].title;
TweenLite.to(_motd, 2.0, {alpha:1.0, onComplete:onMOTDFadeInComplete})
}
}

private
```

```
function onMOTDFadeInComplete():void
{
TweenLite.to(_motd, 2.0, {alpha:0.0, onComplete:onMOTDTweenComplete, delay:10.0})
}

private function onMOTDTweenComplete():void
{
++_currentMOTDIndex;
if (_currentMOTDIndex >= _motdMessages.length || _currentMOTDIndex < 0)
_currentMOTDIndex = 0;

_motd.text = _motdMessages[_currentMOTDIndex].title;
TweenLite.to(_motd, 2.0, {alpha:1.0, onComplete:onMOTDFadeInComplete})
}


//==========================================================================
//**************************************************************************************************
// GENERAL VIEW
//**************************************************************************************************

//==========================================================================

[Inject]
public function set presenter( value:IChatPresenter ):void { _presenter = value; }
public function get presenter():IChatPresenter { return IChatPresenter(_presenter); }

[Inject]
public function set stage( value:Stage ):void { _stage = value; }

[Inject]
public function set tooltip( value:Tooltips ):void { _tooltip = value; }

override public function get height():Number { return _tabs.height * Application.SCALE; }
override public function get width():Number { return _tabs.width * Application.SCALE; }

override public function get type():String { return ViewEnum.UI }

override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
_keyboard.removeKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);
presenter.removeChatListener(onChatUpdate);
presenter.removeMotDUpdatedListener(onNewMOTDMessageAdded);
presenter.removeOnDefaultChannelUpdatedListener(onDefaultChannelUpdated);

super.destroy();

if (_tabs)
{
```

```
_tabs.removeSwitchTabListener(onTabSwitched);
_tabs.destroy();
}

_tabs = null;

if (_chatLog)
_chatLog.destroy();

_chatLog = null;

if (_textInput)
_textInput.destroy();

_textInput = null;

if (_motd)
{
TweenLite.killTweensOf(_motd);
_motd.destroy();
}

_motd = null;

if (_fullBtn)
{
removeListener(_fullBtn, MouseEvent.CLICK, onMouseClick);
_fullBtn.destroy();
}

_fullBtn = null;

if (_halfBtn)
{
removeListener(_halfBtn, MouseEvent.CLICK, onMouseClick);
_halfBtn.destroy();
}

_halfBtn = null;

if (_minBtn)
{
removeListener(_minBtn, MouseEvent.CLICK, onMouseClick);
_minBtn.destroy();
}

_minBtn = null;

if (_ignorePlayerListBtn)
{
```

```
removeListener(_ignorePlayerListBtn, MouseEvent.CLICK, popChatContextMenu);
}

_ignorePlayerListBtn = null;

_alertedBtn = null;

_stage = null;

_inputBG = null;
_chatBG = null;
_motdBG = null;

_motdHolder = null;
_chatHolder = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar = null;

_scrollRect = null;

if (_keyboard)
_keyboard.removeKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);

_keyboard = null;

}

}
}

-----------------------------------
File 741: igw\com\ui\hud\shared\IconDrawerView.as
package com.ui.hud.shared
{
import com.Application;
import com.enum.PositionEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.PanelEnum;
import com.event.StateEvent;
import com.model.battle.BattleRerollVO;
import com.model.player.CurrentUser;
import com.model.warfrontModel.WarfrontVO;
import com.presenter.shared.IUIPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.View;
import
```

```
com.ui.core.component.button.BitmapButton;
import com.ui.core.component.pulldown.DrawerComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.core.effects.EffectFactory;
import com.ui.hud.shared.leaderboards.LeaderboardView;
import com.ui.hud.shared.mail.MailBoxView;
import com.ui.modal.PanelFactory;
import com.ui.modal.alliances.alliance.AllianceView;
import com.ui.modal.alliances.noalliance.NoAllianceView;
import com.ui.modal.battle.chance.GameOfChanceListView;
import com.ui.modal.battlelog.BattleLogListView;
import com.ui.modal.warfront.WarfrontView;

import flash.display.Bitmap;
import flash.events.Event;
import flash.events.MouseEvent;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.parade.core.IView;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class IconDrawerView extends View
{
private const MIN_Y_POS:Number = 602;

private var _btnAlliance:BitmapButton;
private var _btnBattlelog:BitmapButton;
private var _btnLeaderboard:BitmapButton;
private var _btnMail:BitmapButton;
private var _btnWarfront:BitmapButton;
private var _btnChanceGame:BitmapButton;
private var _drawer:DrawerComponent;
private var _mailExclamation:Bitmap;
private var _tooltip:Tooltips;
private var _warfrontGlow:Bitmap;
private var _warfrontPulseCount:uint;
private var _rerollsGlow:Bitmap;
private var _rerollsPulseCount:uint;


private var _mailText:String = 'CodeString.IconDrawer.MailTooltip'; //Mail
private var _battlelogText:String = 'CodeString.IconDrawer.BattleLogTooltip'; //Battle Log
private var _leaderboardText:String = 'CodeString.IconDrawer.LeaderboardTooltip';
//Leaderboard
private var _warfrontText:String = 'CodeString.IconDrawer.WarfrontTooltip'; //Warfront
private var _allianceText:String = 'CodeString.IconDrawer.AllianceTooltip'; //Alliance
private var _pendingScansText:String = 'CodeString.IconDrawer.PendingScansTooltip';
//Pending
```

Scans

```
[PostConstruct]
override public function init():void
{
super.init();
presenter.addMailCountUpdateListener(onMailCountUpdated);
presenter.addWarfrontUpdateListener(onWarfrontUpdated);
presenter.addAvailableRerollUpdatedListener(onRerollsUpdated);

var xpos:Number = 17.5;
var ypos:Number = 1;
_btnWarfront = UIFactory.getButton(ButtonEnum.ICON_WARFRONT, 0, 0, xpos, ypos);
_btnMail = UIFactory.getButton(ButtonEnum.ICON_MAIL, 0, 0, xpos, ypos += 32);
_btnBattlelog = UIFactory.getButton(ButtonEnum.ICON_BATTLE_LOG, 0, 0, xpos, ypos += 22);
_btnLeaderboard = UIFactory.getButton(ButtonEnum.ICON_LEADERBOARD, 0, 0, xpos, ypos
+= 27);
_btnAlliance = UIFactory.getButton(ButtonEnum.ICON_ALLIANCE, 0, 0, xpos, ypos += 22);
_btnChanceGame = UIFactory.getButton(ButtonEnum.ICON_CHANCE, 0, 0, xpos, ypos += 27);

addListener(_btnMail, MouseEvent.CLICK, onButtonClicked);
addListener(_btnBattlelog, MouseEvent.CLICK, onButtonClicked);
addListener(_btnLeaderboard, MouseEvent.CLICK, onButtonClicked);
addListener(_btnWarfront, MouseEvent.CLICK, onButtonClicked);
addListener(_btnAlliance, MouseEvent.CLICK, onButtonClicked);
addListener(_btnChanceGame, MouseEvent.CLICK, onButtonClicked);

//tooltips
var loc:Localization = Localization.instance;
_tooltip.addTooltip(_btnMail, this, null, loc.getString(_mailText));
_tooltip.addTooltip(_btnBattlelog, this, null, loc.getString(_battlelogText));
_tooltip.addTooltip(_btnLeaderboard, this, null, loc.getString(_leaderboardText));
_tooltip.addTooltip(_btnWarfront, this, null, loc.getString(_warfrontText));
_tooltip.addTooltip(_btnAlliance, this, null, loc.getString(_allianceText));
_tooltip.addTooltip(_btnChanceGame, this, null, loc.getString(_pendingScansText));

_drawer = new DrawerComponent();
_drawer.init(PanelEnum.CONTAINER_DOUBLE_NOTCHED_ARROWS, 62, 32,
DrawerComponent.EXPANDS_UP, 15, 35);
_drawer.showInnerPanel = true;
_drawer.addElement(_btnMail);
_drawer.addElement(_btnBattlelog);
_drawer.addElement(_btnLeaderboard);
_drawer.addElement(_btnWarfront);
_drawer.addElement(_btnAlliance);
_drawer.addElement(_btnChanceGame);
_drawer.y = _drawer.height;
_drawer.maximize();

addChild(_drawer);

addHitArea();
```

```
addEffects();
effectsIN();
onStageResized();

visible = !presenter.inFTE;
}

private function onButtonClicked( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;

var view:IView;
switch (e.currentTarget)
{
//mail button
case _btnMail:
//var event:StateEvent = new StateEvent(StateEvent.SHUTDOWN_START);
//presenter.dispatch(event);
view = _viewFactory.createView(MailBoxView);
showHideMailExclamation(false);
break;

//battle log
case _btnBattlelog:
view = _viewFactory.createView(BattleLogListView);
break;

//leaderboard button
case _btnLeaderboard:
view = _viewFactory.createView(LeaderboardView);
break;

//warfront button
case _btnWarfront:
view = _viewFactory.createView(WarfrontView);
break;

//alliance button
case _btnAlliance:
if (CurrentUser.alliance != '')
{
var allianceView:AllianceView = AllianceView(_viewFactory.createView(AllianceView));
allianceView.allianceKey = CurrentUser.alliance;
_viewFactory.notify(allianceView);
} else
{
view = _viewFactory.createView(NoAllianceView);
}
break;
```

```
//chance game button
case _btnChanceGame:
TweenLite.killTweensOf(_rerollsGlow);
_drawer.removeElement(_rerollsGlow);
_rerollsGlow = null;
view = _viewFactory.createView(GameOfChanceListView);
break;
}

if (view)
_viewFactory.notify(view);
}

public function onMailCountUpdated( unread:uint, count:uint, serverUpdate:Boolean ):void
{
showHideMailExclamation(unread > 0);
}

private function showHideMailExclamation( show:Boolean = true ):void
{
if (show && !_mailExclamation)
{
_mailExclamation = UIFactory.getBitmap("ExclamationBMD");
_mailExclamation.x = 18;
_mailExclamation.y = -8;
_btnMail.addChild(_mailExclamation);
} else if (!show && _mailExclamation)
{
_btnMail.removeChild(_mailExclamation);
ObjectPool.give(_mailExclamation);
_mailExclamation = null;
}
}

private function onWarfrontUpdated( battles:Vector.<WarfrontVO>, removed:Vector.<String>
):void
{
if (battles && battles.length > 0)
{
if (!_warfrontGlow)
{
_warfrontGlow = PanelFactory.getPanel('BtnIconWarfrontSelectedBMD');
_warfrontGlow.x = _btnWarfront.x;
_warfrontGlow.y = _btnWarfront.y;
_drawer.addElement(_warfrontGlow);
}
TweenLite.killTweensOf(_warfrontGlow);
_warfrontPulseCount = 0;
onWarfrontGlowFadeOut();
```

```
}
}

private function onWarfrontGlowFadeOut():void
{
if (_warfrontPulseCount < 4 && _warfrontGlow != null)
{
++_warfrontPulseCount;
TweenLite.to(_warfrontGlow, 1.0, {alpha:1.0, ease:Quad.easeOut,
onComplete:onWarfrontGlowFadeIn, overwrite:0});
} else
{
_warfrontPulseCount = 0;
if (_warfrontGlow)
{
TweenLite.killTweensOf(_warfrontGlow);
_drawer.removeElement(_warfrontGlow);
}
_warfrontGlow = null;
}
}

private function onWarfrontGlowFadeIn():void
{
if (_warfrontPulseCount < 4 && _warfrontGlow != null)
{
++_warfrontPulseCount;
TweenLite.to(_warfrontGlow, 1.0, {alpha:0.0, ease:Quad.easeIn,
onComplete:onWarfrontGlowFadeOut, overwrite:0});
} else
{
_warfrontPulseCount = 0;
if (_warfrontGlow)
{
TweenLite.killTweensOf(_warfrontGlow);
_drawer.removeElement(_warfrontGlow);
}
_warfrontGlow = null;
}
}

private function onRerollsUpdated( battleReroll:BattleRerollVO ):void
{
if (battleReroll)
{
if (!_rerollsGlow)
{
_rerollsGlow = PanelFactory.getPanel('ExclamationBMD');
_rerollsGlow.x = _btnChanceGame.x + (_btnChanceGame.width * 0.5) - (_rerollsGlow.width *
0.5);
```

```
_rerollsGlow.y = _btnChanceGame.y + (_btnChanceGame.height * 0.5) - (_rerollsGlow.height *
0.5);
_drawer.addElement(_rerollsGlow);
}
TweenLite.killTweensOf(_rerollsGlow);
_rerollsPulseCount = 0;
onRerollsGlowFadeOut();
}
}

private function onRerollsGlowFadeOut():void
{
TweenLite.to(_rerollsGlow, 1.0, {alpha:1.0, ease:Quad.easeOut,
onComplete:onRerollsGlowFadeIn, overwrite:0});
}

private function onRerollsGlowFadeIn():void
{
TweenLite.to(_rerollsGlow, 1.0, {alpha:0.0, ease:Quad.easeIn,
onComplete:onRerollsGlowFadeOut, overwrite:0});
}

override protected function onStateChange( state:String ):void
{
if (state == StateEvent.GAME_BATTLE)
destroy();
}

override protected function addEffects():void {
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.BOTTOM,
onStageResized)); }

private function onStageResized( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
x = 523 * Application.SCALE;
y = DeviceMetrics.HEIGHT_PIXELS;
y = (y < MIN_Y_POS) ? MIN_Y_POS : y;
}

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

[Inject]
public function set tooltip( value:Tooltips ):void { _tooltip = value; }

override public function get type():String { return ViewEnum.UI }
override public function get screenshotBlocker():Boolean {return true;}
override
```

```
public function destroy():void
{
presenter.removeMailCountUpdateListener(onMailCountUpdated);
presenter.removeWarfrontUpdateListener(onWarfrontUpdated);
presenter.removeAvailableRerollUpdatedListener(onRerollsUpdated);

super.destroy();

showHideMailExclamation(false);

_btnAlliance = UIFactory.destroyButton(_btnAlliance);
_btnBattlelog = UIFactory.destroyButton(_btnBattlelog);
_btnLeaderboard = UIFactory.destroyButton(_btnLeaderboard);
_btnMail = UIFactory.destroyButton(_btnMail);
_btnWarfront = UIFactory.destroyButton(_btnWarfront);

ObjectPool.give(_drawer);
_drawer = null;

if (_warfrontGlow)
TweenLite.killTweensOf(_warfrontGlow);
if (_rerollsGlow)
TweenLite.killTweensOf(_rerollsGlow);
_rerollsGlow = null;
_tooltip.removeTooltip(null, this);
_tooltip = null;
}
}
}
```

File 742: igw\com\ui\hud\shared\MiniMapView.as

```
package com.ui.hud.shared
{
import com.Application;
import com.enum.CategoryEnum;
import com.enum.PositionEnum;
import com.enum.ToastEnum;
import com.enum.TypeEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.StateEvent;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.shared.Owned;
import com.game.entity.components.shared.Position;
import com.google.analytics.debug.Panel;
import com.model.mission.MissionVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import
```

```
com.presenter.sector.IMiniMapPresenter;
import com.presenter.shared.IUIPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.core.effects.EffectFactory;
import com.ui.hud.sector.bookmarks.BookmarksView;
import com.ui.modal.settings.SettingsView;
import com.util.AllegianceUtil;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Shape;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.filters.ColorMatrixFilter;
import flash.geom.ColorTransform;
import flash.geom.Point;
import flash.geom.Rectangle;
import flash.globalization.DateTimeFormatter;
import flash.globalization.DateTimeStyle;
import flash.globalization.LocaleID;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.Timer;
import flash.utils.getDefinitionByName;

import org.ash.core.Entity;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class MiniMapView extends View
{
/** For the minimap */
private var _bg:Bitmap;
private var _battery:Bitmap;
private var _utilityBG:ScaleBitmap;
private var _target:Sprite;
private var _mouseTarget:Sprite;
private var _ownedShipLayer:Sprite;
private var _nonOwnedShipLayer:Sprite;
private
```

```
var _defaultLayer:Sprite;
private var _bezel:Sprite;
private var _currentTime:Label;
private var _coordLabel:Label;
private var _batteryLife:ProgressBar;

private var _bookmarksButton:BitmapButton;
private var _exitCombatButton:BitmapButton;
private var _findButton:BitmapButton;
private var _maximizeButton:BitmapButton;
private var _minimizeButton:BitmapButton;
private var _missionButton:BitmapButton;
private var _retreatButton:BitmapButton;
private var _instancedMissionButton:BitmapButton;

private var _settingsButton:BitmapButton;
private var _switchStateButton:BitmapButton;

private var _redFilter:ColorMatrixFilter;
private var _orangeFilter:ColorMatrixFilter;
private var _yellowFilter:ColorMatrixFilter;
private var _greenFilter:ColorMatrixFilter;
private var _currentFilter:ColorMatrixFilter;

private var _uiPresenter:IUIPresenter;

private var _icons:Dictionary;
private var _bounds:Rectangle;
private var _mouseDownPoint:Point;
private var _isDragging:Boolean;
private var _isDraggingBezel:Boolean;
private var _tmpBuildingIcon:BitmapData;
private var _batteryBMD:BitmapData;
private var _ChargingBatteryBMD:BitmapData;
private var _currentDate:Date;
private var _formatter:DateTimeFormatter;
private var _timer:Timer;

private var _tooltip:Tooltips;

private const TARGET_OFFSET_X:Number = 7;
private const TARGET_OFFSET_Y:Number = 29;

private const BEZEL_ROT_OFFSET:Number = -45;
private const BEZEL_ROT_RANGE:Number = 90;

private const MIN_X_POS:Number = 1060;

private var _enterBase:String = 'CodeString.Controls.ViewBase'; //VIEW BASE
private var _exitBase:String = 'CodeString.Controls.ViewMap'; //VIEW MAP
private
```

```
var _enterInstancedMission:String = 'CodeString.Controls.EnterInstancedMission'; //ENTER
INSTANCED MISSION
private var _retreat:String = 'CodeString.Controls.Retreat'; //RETREAT
private var _findBase:String = 'CodeString.Controls.FindBase'; //Find Starbase
private var _fullScreen:String = 'CodeString.Controls.FullScreen'; //Fullscreen
private var _bookmarks:String = 'CodeString.Bookmarks.Title'; //Bookmarks
private var _minimize:String = 'CodeString.Controls.Minimize'; //Minimize
private var _gotoMission:String = 'CodeString.Controls.GotoMission'; //Go to Mission
private var _settings:String = 'CodeString.Controls.Settings'; //Settings

[PostConstruct]
override public function init():void
{
super.init();

_currentFilter = _greenFilter = CommonFunctionUtil.getColorMatrixFilter(0x3cf219);
_yellowFilter = CommonFunctionUtil.getColorMatrixFilter(0xfbe81a);
_orangeFilter = CommonFunctionUtil.getColorMatrixFilter(0xfa7d0e);
_redFilter = CommonFunctionUtil.getColorMatrixFilter(0xf81919);

_batteryBMD = UIFactory.getBitmapData("BatteryBMD");
_ChargingBatteryBMD = UIFactory.getBitmapData("ChargingBatteryBMD");

_currentDate = new Date();
_formatter = new DateTimeFormatter(Application.COUNTRY);
_formatter.setDateTimePattern('hh:mm a');

var timeToNextMin:Number = (60 - _currentDate.seconds) * 1000;

_timer = new Timer(timeToNextMin, 1)
_timer.addEventListener(TimerEvent.TIMER_COMPLETE, onFirstTimerTick, false, 0, true);
_timer.start();

_icons = new Dictionary();

presenter.addToMiniMapSignal.add(addEntityToMiniMap);
presenter.clearMiniMapSignal.add(clearMiniMap);
presenter.removeFromMiniMapSignal.add(removeEntityFromMiniMap);
presenter.scrollMiniMapSignal.add(repositionAllIcons);

_bg = UIFactory.getBitmap("MinimapBMD");
_bg.y = 21;
_bg.smoothing = true;
addChild(_bg);

_utilityBG = UIFactory.getPanel(PanelEnum.CONTAINER_INNER_DARK, 168, 25, 0, -8);
addChild(_utilityBG);

_batteryLife = ObjectPool.get(ProgressBar);
_batteryLife.init(ProgressBar.VERTICAL, UIFactory.getPanel(PanelEnum.STATBAR_GREY,
11,
```

```
17), null);
_batteryLife.setMinMax(0, 1);
_batteryLife.amount = 0;
_batteryLife.x = _utilityBG.x + 6;
_batteryLife.y = _utilityBG.y + 5;
_batteryLife.filters = [_currentFilter];
_batteryLife.amount = Application.batteryLife;
_batteryLife.visible = false;
addChild(_batteryLife);

_battery = new Bitmap();
_battery.x = _utilityBG.x + 5;
_battery.y = _utilityBG.y + 2;
_battery.visible = false;

if (Application.isCharging)
_battery.bitmapData = _ChargingBatteryBMD;
else
_battery.bitmapData = _batteryBMD;

addChild(_battery);

_currentTime = new Label(20, 0xacd1ff);
_currentTime.constrictTextToSize = false;
_currentTime.align = TextFormatAlign.CENTER;
updateClock();
addChild(_currentTime);

_currentDate.time += timeToNextMin;

var targetClass:Class = Class(getDefinitionByName('MiniMapTargetMC'));
var maskClass:Class = Class(getDefinitionByName('MiniMapMaskMC'));

_target = Sprite(new targetClass());
var mask:Sprite = Sprite(new maskClass());

mask.x = TARGET_OFFSET_X;
mask.y = TARGET_OFFSET_Y;
mask.cacheAsBitmap = true;
addChild(mask);
_target.mask = mask;

_defaultLayer = new Sprite();
_target.addChild(_defaultLayer);

_nonOwnedShipLayer = new Sprite();
_target.addChild(_nonOwnedShipLayer);

_ownedShipLayer = new Sprite();
_target.addChild(_ownedShipLayer);

_target.x
```

```
= TARGET_OFFSET_X;
_target.y = TARGET_OFFSET_Y;
_target.cacheAsBitmap = true;
addChildAt(_target, numChildren - 1);

// We grab this at init because the target's dimensions can change when its children (i.e. the icons) move beyond the visible bounds.
_bounds = new Rectangle(_target.x, _target.y, _target.width, _target.height);
presenter.miniMapWidth = _target.width;

// Cover the minimap with a slightly visible sprite to catch mouse events
_mouseTarget = new Sprite();
_mouseTarget.graphics.beginFill(0xffffff, 0.1);
_mouseTarget.graphics.drawCircle(84, 106, 80);
_mouseTarget.graphics.endFill();
addChild(_mouseTarget);

addListener(_mouseTarget, MouseEvent.MOUSE_DOWN, onMouseDown);
addListener(_mouseTarget, MouseEvent.MOUSE_UP, onMouseUp);
addListener(_mouseTarget, MouseEvent.MOUSE_MOVE, onMouseMove);
addListener(_mouseTarget, MouseEvent.MOUSE_WHEEL, onMouseWheel);
addListener(_mouseTarget, MouseEvent.MOUSE_OUT, onMouseOut);

// The bezel, where clicking & dragging the bead can change the zoom level
_bezel = new Sprite();
// Spaces out the bezel
_bezel.graphics.beginFill(0, 0);
_bezel.graphics.drawRect(0, 0, mask.width >> 1, mask.height >> 1);
_bezel.graphics.endFill();

var bead:Sprite = new Sprite();
var beadBmp:Bitmap = UIFactory.getBitmap('ZoomHandleBMD');
beadBmp.smoothing = true;
bead.addChild(beadBmp);
bead.x = (mask.width - bead.width) / 2 + 1;
bead.y = -bead.height / 2;
addListener(bead, MouseEvent.MOUSE_DOWN, onMouseDownBezel);
addListener(bead, MouseEvent.MOUSE_UP, onMouseUpBezel);

_bezel.addChild(bead);
_bezel.x = TARGET_OFFSET_X + (mask.width >> 1);
_bezel.y = TARGET_OFFSET_Y + (mask.height >> 1) + 3;
_bezel.mouseEnabled = false;
_bezel.mouseChildren = true;
addChild(_bezel);
rotateBezel();

_coordLabel = UIFactory.getLabel(LabelEnum.H4, _bg.width, 30, 0, _bg.y + _bg.height + 4);
_coordLabel.textColor = 0xecffff;
_coordLabel.useLocalization = false;
addChild(_coordLabel);
```

```
addListener(stage, MouseEvent.MOUSE_MOVE, onMouseMoveStage);
addListener(stage, MouseEvent.MOUSE_UP, onMouseUpStage);

_bookmarksButton = UIFactory.getButton(ButtonEnum.ICON_BOOKMARKS, 0, 0, -7, 157);
_exitCombatButton = UIFactory.getButton(ButtonEnum.RED_A, _bg.width, 40, 0, _bg.y +
_bg.height + 16, _exitBase);
_findButton = UIFactory.getButton(ButtonEnum.ICON_FIND, 0, 0, -25, 128);
_maximizeButton = UIFactory.getButton(ButtonEnum.ICON_MAXIMIZE, 0, 0, -6, 21);
_minimizeButton = UIFactory.getButton(ButtonEnum.ICON_MINIMIZE, 0, 0, -6, 21);
_missionButton = UIFactory.getButton(ButtonEnum.ICON_MISSION, 0, 0, -33, 95);
_retreatButton = UIFactory.getButton(ButtonEnum.RED_A, _bg.width, 40, 0,
_exitCombatButton.y + _exitCombatButton.height + 4, _retreat);
_settingsButton = UIFactory.getButton(ButtonEnum.ICON_SETTINGS, 0, 0, 142, 21);
_switchStateButton = UIFactory.getButton(ButtonEnum.BLUE_A, _bg.width, 40, 0, _bg.y +
_bg.height + 16, _exitBase);
_instancedMissionButton = UIFactory.getButton(ButtonEnum.RED_A, _bg.width, 40, 0,
_switchStateButton.y + _switchStateButton.height + 4, _enterInstancedMission);

addButton(_bookmarksButton);
addButton(_exitCombatButton, true);
removeChild(_exitCombatButton);
addButton(_findButton);
addButton(_maximizeButton);
addButton(_minimizeButton);
addButton(_missionButton);
addButton(_retreatButton, true);
removeChild(_retreatButton);
addButton(_settingsButton);
addButton(_switchStateButton, true);
addButton(_instancedMissionButton, true);
removeChild(_instancedMissionButton);


var Loc:Localization = Localization.instance;

_tooltip.addTooltip(_bookmarksButton, this, null, Loc.getString(_bookmarks));
_tooltip.addTooltip(_findButton, this, null, Loc.getString(_findBase));
_tooltip.addTooltip(_maximizeButton, this, null, Loc.getString(_fullScreen));
_tooltip.addTooltip(_minimizeButton, this, null, Loc.getString(_minimize));
_tooltip.addTooltip(_missionButton, this, null, Loc.getString(_gotoMission));
_tooltip.addTooltip(_settingsButton, this, null, Loc.getString(_settings));

Application.onBatteryChargeChanged.add(updateBatteryLife);

addHitArea();
addEffects();
effectsIN();
onStageResize();
onStateChange(Application.STATE);
```

```
visible = !presenter.inFTE;
}

private function clickPreventer( e:Event ):void
{
e.preventDefault();
e.stopImmediatePropagation();
}

private function addEntityToMiniMap( entity:Entity, quadBounds:Rectangle ):void
{
// Entities can straddle multiple grid cells, so we can get multiple add notifs.
// Only add them the first time we encounter the entity ID, and ignore subsequent ones.
if (_icons.hasOwnProperty(entity.id))
{
return;
}

var detail:Detail = entity.get(Detail);
if (!detail)
return;

var isTmpIcon:Boolean;
var iconClass:Class;
var layer:Sprite = _defaultLayer;
var transgateColor:ColorTransform;

switch (detail.category)
{
case CategoryEnum.SHIP:
iconClass = Class(getDefinitionByName('MiniMapFleetIconBMD'));
if (entity.has(Owned))
{
layer = _ownedShipLayer;
} else
{
layer = _nonOwnedShipLayer;
}
break;

case CategoryEnum.SECTOR:
switch (detail.type)
{
case TypeEnum.DERELICT_IGA:
case TypeEnum.DERELICT_SOVEREIGNTY:
case TypeEnum.DERELICT_TYRANNAR:
iconClass = Class(getDefinitionByName("MiniMapCargoIconBMP"));
break;

case
```

```
TypeEnum.TRANSGATE_IGA:
transgateColor = AllegianceUtil.CT_IGA_BASE;
// intentional fall-through
case TypeEnum.TRANSGATE_SOVEREIGNTY:
transgateColor ||= AllegianceUtil.CT_SOVEREIGNTY_BASE;
// intentional fall-through
case TypeEnum.TRANSGATE_TYRANNAR:
transgateColor ||= AllegianceUtil.CT_TYRANNAR_BASE;
iconClass = Class(getDefinitionByName('SectorFleetTransgateIconBMD'));
break;

default:
iconClass = Class(getDefinitionByName('MiniMapBaseIconBMD'));
}
break;

case CategoryEnum.BUILDING:
{
// iconClass = Class(getDefinitionByName("MiniMapBaseIconBMD"));
//PR Disabling for the moment. Showing buildings on the minimap is causing a crash when
making new
//isTmpIcon = true;

break;
}

default:
// If it's not one of the listed types, it doesn't show up on the minimap.
}

if (iconClass || isTmpIcon)
{
if (iconClass)
{
var bmd:BitmapData = BitmapData(new iconClass());
var icon:Bitmap = new Bitmap(bmd);
}

else
icon = new Bitmap(TEMP_buildingIcon);

if (transgateColor)
icon.transform.colorTransform = transgateColor;

else
{
var useOwnerColors:Boolean;
if (detail.type == TypeEnum.STARBASE_SECTOR_IGA || detail.type ==
TypeEnum.STARBASE_SECTOR_SOVEREIGNTY || detail.type ==
TypeEnum.STARBASE_SECTOR_TYRANNAR)
{
```

```
var selectedEntityPlayer:PlayerVO = presenter.getPlayer(detail.ownerID)
var baseAttackRatingDifferenceLimit:int =
presenter.getConstantPrototypeByName('baseAttackRatingDifferenceLimit');
var baseAttackFreeForAllRatingThreshold:int =
presenter.getConstantPrototypeByName('baseAttackFreeForAllRatingThreshold');
var focusedFleetLevel:int = presenter.focusedFleetRating;
var diff:int = focusedFleetLevel - detail.baseLevel;

if (selectedEntityPlayer && !selectedEntityPlayer.isNPC && CurrentUser.faction !=
selectedEntityPlayer.faction)
{
if (diff > baseAttackRatingDifferenceLimit && detail.baseLevel <
baseAttackFreeForAllRatingThreshold)
useOwnerColors = false;
else
useOwnerColors = true;
}
}

if (detail.ownerID == CurrentUser.id)
useOwnerColors = true;

icon.transform.colorTransform = AllegianceUtil.instance.getEntityColorTransform(entity,
useOwnerColors);
}

layer.addChild(icon);

var position:Position = entity.get(Position);
positionIcon(icon, position);

var scaleFactor:Number = getScaleFactor();
(icon as Bitmap).scaleX = scaleFactor;
(icon as Bitmap).scaleY = scaleFactor;

_icons[entity.id] = icon;
_icons[icon] = entity.id;
}
}

private function onButtonClick( e:MouseEvent ):void
{
e.preventDefault();
e.stopPropagation();
e.stopImmediatePropagation();
if (!presenter.hudEnabled)
return;

switch (e.currentTarget)
{
```

```
case _bookmarksButton:
if (!presenter.fteRunning)
showView(BookmarksView);
break;
case _exitCombatButton:
presenter.showSector();
break;
case _findButton:
presenter.findBase(CurrentUser.id);
break;
case _maximizeButton:
_maximizeButton.visible = false;
_minimizeButton.visible = true;
_uiPresenter.toggleFullScreen();
break;
case _minimizeButton:
_maximizeButton.visible = true;
_minimizeButton.visible = false;
_uiPresenter.toggleFullScreen();
break;
case _missionButton:
var result:String = presenter.moveToMissionTarget();
if (result)
showToast(ToastEnum.WRONG, null, result);
break;
case _retreatButton:
presenter.retreat();
break;
case _settingsButton:
if (!presenter.fteRunning)
showView(SettingsView);
break;
case _switchStateButton:
if (Application.STATE == StateEvent.GAME_STARBASE)
presenter.showSector();
else if (Application.STATE == StateEvent.GAME_SECTOR)
presenter.enterStarbase();
break;
case _instancedMissionButton:
presenter.enterInstancedMission();
break
}
}

private function get TEMP_buildingIcon():BitmapData
{
if (!_tmpBuildingIcon)
{
var s:Shape = new Shape();
s.graphics.beginFill(0xf0f0f0,
```

```
0.5);
s.graphics.moveTo(10, 0);
s.graphics.lineTo(0, 5);
s.graphics.lineTo(10, 10);
s.graphics.lineTo(20, 5);
s.graphics.lineTo(10, 0);
s.graphics.endFill();

_tmpBuildingIcon = new BitmapData(s.width, s.height, true, 0x00000000);
_tmpBuildingIcon.draw(s);
}

return _tmpBuildingIcon;
}

private function removeEntityFromMiniMap( entity:Entity ):void
{
var icon:Bitmap = _icons[entity.id];
if (icon)
{
icon.parent.removeChild(icon);
}

delete _icons[entity.id];
delete _icons[icon];
}

private function clearMiniMap():void
{
for each (var key:* in _icons)
{
if (key is Bitmap)
{
(key as Bitmap).parent.removeChild(key);
}
}

_icons = new Dictionary();
}

private function getScaleFactor():Number
{
var scaleFactor:Number = presenter.zoom;
scaleFactor = scaleFactor < 0.5 ? 0.5 : scaleFactor;
scaleFactor = scaleFactor > 1.5 ? 1.5 : scaleFactor;
return scaleFactor;
}

private function repositionAllIcons():void
{
if
```

```
(_icons)
{
var scaleFactor:Number = getScaleFactor();
for (var icon:* in _icons)
{
if (icon)
{
if (icon is Bitmap)
{
var id:String = _icons[icon];
var entity:Entity = presenter.getEntity(id);
if (entity)
{
var position:Position = entity.get(Position) as Position;
if (position)
{
positionIcon(icon, position);

(icon as Bitmap).scaleX = scaleFactor;
(icon as Bitmap).scaleY = scaleFactor;
}
}
}
}
}
}
}

private function positionIcon( icon:Bitmap, position:Position ):void
{
var coords:Point = presenter.getIconPosition(_bounds.width, _bounds.height, position);

icon.x = coords.x - (icon.width >> 1);
icon.y = coords.y - (icon.height >> 1);
}

override protected function onMouseDown( event:MouseEvent ):void
{
if (event.target == _mouseTarget)
{
var hit:Point = new Point(event.localX, event.localY);
_mouseDownPoint = hit;
_isDragging = true;
presenter.mouseDown();
event.stopImmediatePropagation();
}
}

override protected function onRightMouse( event:MouseEvent ):void
{
if
```

```
(event.target == _mouseTarget)
event.stopImmediatePropagation();
}

private function onMouseUp( event:MouseEvent ):void
{
if (_isDraggingBezel)
{
_isDragging = _isDraggingBezel = false;
return;
}
_isDragging = false;
presenter.mouseUp();

if (!_mouseDownPoint)
return;

var xDelta:Number = Math.abs(event.localX - _mouseDownPoint.x);
var yDelta:Number = Math.abs(event.localY - _mouseDownPoint.y);

if (xDelta + yDelta <= 3)
{
presenter.mouseMove((_bounds.width >> 1) - _mouseDownPoint.x, (_bounds.height >> 1) -
_mouseDownPoint.y);
}
}

private function onMouseMove( event:MouseEvent ):void
{
if (_isDragging && !_isDraggingBezel)
{
presenter.mouseMove(event.localX - _mouseDownPoint.x, event.localY - _mouseDownPoint.y);
}
}

private function onMouseWheel( event:MouseEvent ):void
{
presenter.mouseWheel(event.delta);
repositionAllIcons();
rotateBezel();
event.stopImmediatePropagation();
}

private function onMouseOut( event:MouseEvent ):void
{
_isDragging = false;
}

private function onMouseDownBezel( event:MouseEvent ):void
{
event.stopImmediatePropagation();
```

```
_isDraggingBezel = true;
event.stopImmediatePropagation();
}

private function onMouseUpBezel( event:MouseEvent ):void
{
event.stopImmediatePropagation();
_isDragging = _isDraggingBezel = false;
}

private function onMouseMoveStage( event:MouseEvent ):void
{
if (_isDraggingBezel)
{
var pt:Point = _bezel.localToGlobal(new Point(0, 0));
var angle:Number = Math.atan2(event.stageY - pt.y, event.stageX - pt.x);
var angleDeg:Number = (angle / Math.PI) * 180;
presenter.zoomPercent = (angleDeg + BEZEL_ROT_OFFSET) / BEZEL_ROT_RANGE;
repositionAllIcons();
rotateBezel();
}
}

private function onMouseUpStage( event:MouseEvent ):void
{
event.stopImmediatePropagation();
_isDragging = _isDraggingBezel = false;
}

private function rotateBezel():void
{
// Rotation in degrees
_bezel.rotation = (presenter.zoomPercent * BEZEL_ROT_RANGE) - BEZEL_ROT_OFFSET;
}

override protected function onStateChange( type:String ):void
{
presenter.removeListenerToUpdateMission(updateMissionButton);
switch (type)
{
case StateEvent.GAME_BATTLE:
_bookmarksButton.enabled = _findButton.enabled = _missionButton.enabled = false;
_coordLabel.visible = _switchStateButton.visible = false;
if (presenter.isMissionBattle)
{
_exitCombatButton.text = _retreat;
if (presenter.showRetreat)
{
_retreatButton.y = _exitCombatButton.y;
addChild(_retreatButton);
```

```
} else
addChild(_exitCombatButton);
} else
{
_exitCombatButton.text = _exitBase;
addChild(_exitCombatButton);
if (presenter.showRetreat)
{
_retreatButton.y = _exitCombatButton.y + _exitCombatButton.height + 4
addChild(_retreatButton);
}
}

if(contains(_instancedMissionButton))
removeChild(_instancedMissionButton);

break;
case StateEvent.GAME_SECTOR:
updateMissionButton();
_bookmarksButton.enabled = _findButton.enabled = true;
_switchStateButton.text = _enterBase;
_coordLabel.visible = _switchStateButton.visible = true;
_switchStateButton.y = _bg.y + _bg.height + 30;
if (contains(_exitCombatButton))
removeChild(_exitCombatButton);
if (contains(_retreatButton))
removeChild(_retreatButton);

if(presenter.isInInstancedMission())
{
_instancedMissionButton.y = _switchStateButton.y + _switchStateButton.height + 4;
if(!contains(_instancedMissionButton))
{
addChild(_instancedMissionButton);
}
}
else
{
if(contains(_instancedMissionButton))
removeChild(_instancedMissionButton);
}

presenter.removeListenerOnCoordsUpdate(onCoordsUpdated);
presenter.addListenerOnCoordsUpdate(onCoordsUpdated);
presenter.removeSelectionChangeListener(onSelectionChange);
presenter.addSelectionChangeListener(onSelectionChange);
presenter.addListenerToUpdateMission(updateMissionButton);
break;
case StateEvent.GAME_STARBASE:
_coordLabel.visible
```

```
= false;
_bookmarksButton.enabled = true;
_findButton.enabled = _missionButton.enabled = false;
_switchStateButton.text = _exitBase;
_switchStateButton.visible = true;
_switchStateButton.y = _bg.y + _bg.height + 16;
if (contains(_exitCombatButton))
removeChild(_exitCombatButton);
if (contains(_retreatButton))
removeChild(_retreatButton);

if(presenter.isInInstancedMission())
{
_instancedMissionButton.y = _switchStateButton.y + _switchStateButton.height + 4;
if(!contains(_instancedMissionButton))
{
addChild(_instancedMissionButton);
}
}
else
{
if(contains(_instancedMissionButton))
removeChild(_instancedMissionButton);
}
break;
}
}

private function onCoordsUpdated( x:int, y:int ):void
{
_coordLabel.text = Localization.instance.getString(presenter.sectorName) + " " +
Localization.instance.getString(presenter.sectorEnum) +
" - " + int(x * 0.01) + "x" + int(y * 0.01) + "";
}

override protected function addEffects():void {
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.RIGHT, PositionEnum.TOP,
onStageResize, x, y)); }

private function onStageResize():void
{
this.scaleX = this.scaleY = Application.SCALE;
var pos:Number = MIN_X_POS * Application.SCALE;
x = (DeviceMetrics.WIDTH_PIXELS - (_bg.width + 12) * Application.SCALE < pos) ? pos :
DeviceMetrics.WIDTH_PIXELS - (12 + _bg.width) * Application.SCALE;
y = 11 * Application.SCALE;

_maximizeButton.visible = !_uiPresenter.isFullScreen;
_minimizeButton.visible = _uiPresenter.isFullScreen;
}

private
```

```
function addButton( btn:BitmapButton, onlyListeners:Boolean = false ):void
{
if (!onlyListeners)
{
var icon:Bitmap = Bitmap(btn.getChildAt(0));
var bitmap:Bitmap = UIFactory.getBitmap("IconCircleBMD");
bitmap.smoothing = true;
btn.addChildAt(bitmap, 0);
icon.x = (bitmap.width - icon.width) * .5;
icon.y = (bitmap.height - icon.height) * .5;
}
addListener(btn, MouseEvent.CLICK, onButtonClick);
addListener(btn, MouseEvent.MOUSE_DOWN, clickPreventer);
addListener(btn, MouseEvent.MOUSE_UP, clickPreventer);
addChild(btn);
}

private function onSelectionChange():void
{
for (var icon:* in _icons)
{
if (icon is Bitmap)
{
var id:String = _icons[icon];
var entity:Entity = presenter.getEntity(id);
if (entity)
{
var detail:Detail = entity.get(Detail);
if (!detail)
continue;

if (detail.type != TypeEnum.TRANSGATE_IGA && detail.type !=
TypeEnum.TRANSGATE_SOVEREIGNTY && detail.type !=
TypeEnum.TRANSGATE_TYRANNAR)
{
var useOwnerColors:Boolean = false;
if (detail.type == TypeEnum.STARBASE_SECTOR_IGA || detail.type ==
TypeEnum.STARBASE_SECTOR_SOVEREIGNTY || detail.type ==
TypeEnum.STARBASE_SECTOR_TYRANNAR)
{
var selectedEntityPlayer:PlayerVO = presenter.getPlayer(detail.ownerID)
var baseAttackRatingDifferenceLimit:int =
presenter.getConstantPrototypeByName('baseAttackRatingDifferenceLimit');
var baseAttackFreeForAllRatingThreshold:int =
presenter.getConstantPrototypeByName('baseAttackFreeForAllRatingThreshold');
var focusedFleetLevel:int = presenter.focusedFleetRating;
var diff:int = focusedFleetLevel - detail.baseLevel;

if (selectedEntityPlayer && !selectedEntityPlayer.isNPC && CurrentUser.faction !=
selectedEntityPlayer.faction)
{
```

```
if (diff > baseAttackRatingDifferenceLimit && detail.baseLevel <
baseAttackFreeForAllRatingThreshold)
useOwnerColors = false;
else
useOwnerColors = true;
}
}

if (detail.ownerID == CurrentUser.id)
useOwnerColors = true;

(icon as Bitmap).transform.colorTransform =
AllegianceUtil.instance.getEntityColorTransform(entity, useOwnerColors);
}
}
}
}
}

private function updateMissionButton():void
{
var currentMission:MissionVO = presenter.currentMission;
if (currentMission)
{
_missionButton.enabled = (currentMission.accepted && currentMission.complete) ? false : true;
} else
_missionButton.enabled = false;
}

private function onFirstTimerTick( e:TimerEvent ):void
{
_currentDate = new Date();
updateClock();
_timer.removeEventListener(TimerEvent.TIMER_COMPLETE, onFirstTimerTick);
_timer.reset();
_timer.repeatCount = 0;
_timer.delay = 10000;
_timer.addEventListener(TimerEvent.TIMER, onTimerTick, false, 0, true);
_timer.start();
}

private function onTimerTick( e:TimerEvent ):void
{
_currentDate = new Date();
updateClock();
}

private function updateClock():void
{
_currentTime.text
```

```
= _formatter.format(_currentDate);
_currentTime.x = _utilityBG.x + (_utilityBG.width - _currentTime.width) * 0.5;
_currentTime.y = _utilityBG.y + (_utilityBG.height - _currentTime.textHeight) * 0.5;
}

public function updateBatteryLife( charging:Boolean, v:Number ):void
{
if (!_battery.visible)
_battery.visible = true;

if (!_batteryLife.visible)
_batteryLife.visible = true;

if (charging)
_battery.bitmapData = _ChargingBatteryBMD;
else
_battery.bitmapData = _batteryBMD;

_batteryLife.amount = v;
var newFilter:ColorMatrixFilter = getBatteryColor(v);
if (_currentFilter != newFilter)
{
_currentFilter = newFilter
_batteryLife.filters = [_currentFilter];
}
}

private function getBatteryColor( batteryLife:Number ):ColorMatrixFilter
{

if (batteryLife >= 0.75)
return _greenFilter;
else if (batteryLife >= 0.50)
return _yellowFilter;
else if (batteryLife >= 0.25)
return _orangeFilter;
else
return _redFilter;
}

override public function get height():Number { return _bg ? _bg.height * Application.SCALE :
this.height * Application.SCALE; }
override public function get width():Number { return _bg ? _bg.width * Application.SCALE :
this.width * Application.SCALE; }

[Inject]
public function set presenter( value:IMiniMapPresenter ):void { _presenter = value; }
public function get presenter():IMiniMapPresenter { return IMiniMapPresenter(_presenter); }

[Inject]
public
```

```
function set uiPresenter( v:IUIPresenter ):void { _uiPresenter = v; }

[Inject]
public function set tooltip( value:Tooltips ):void { _tooltip = value; }

override public function get screenshotBlocker():Boolean {return true;}

override public function get type():String { return ViewEnum.UI }

override public function destroy():void
{
if (_timer.running)
{
_timer.stop();
_timer.removeEventListener(TimerEvent.TIMER, onTimerTick);
}

_timer = null;

Application.onBatteryChargeChanged.remove(updateBatteryLife);

presenter.removeListenerToUpdateMission(updateMissionButton);
presenter.addToMiniMapSignal.remove(addEntityToMiniMap);
presenter.removeFromMiniMapSignal.remove(removeEntityFromMiniMap);
presenter.scrollMiniMapSignal.remove(repositionAllIcons);

removeListener(_mouseTarget, MouseEvent.MOUSE_DOWN, onMouseDown);
removeListener(_mouseTarget, MouseEvent.MOUSE_UP, onMouseUp);
removeListener(_mouseTarget, MouseEvent.MOUSE_MOVE, onMouseMove);
removeListener(_mouseTarget, MouseEvent.MOUSE_WHEEL, onMouseWheel);
removeListener(_mouseTarget, MouseEvent.MOUSE_OUT, onMouseOut);

_tooltip.removeTooltip(null, this);
_tooltip = null;

super.destroy();

_bg = null;
_battery = null;
_utilityBG = null;
_target = null;
_mouseTarget = null;
_ownedShipLayer = null;
_nonOwnedShipLayer = null;
_defaultLayer = null;
_bezel = null;

if (_currentTime)
_currentTime.destroy();

_currentTime
```

```
= null;

if (_coordLabel)
_coordLabel.destroy();

_coordLabel = null;

if (_batteryLife)
ObjectPool.give(_batteryLife);

_coordLabel = null;

if (_bookmarksButton)
_bookmarksButton.destroy();

_bookmarksButton = null;

if (_exitCombatButton)
_exitCombatButton.destroy();

_exitCombatButton = null;

if (_findButton)
_findButton.destroy();

_findButton = null;

if (_maximizeButton)
_maximizeButton.destroy();

_maximizeButton = null;

if (_minimizeButton)
_minimizeButton.destroy();

_minimizeButton = null;

if (_missionButton)
_missionButton.destroy();

_missionButton = null;

if (_retreatButton)
_retreatButton.destroy();

if(_instancedMissionButton)
_instancedMissionButton.destroy();

_instancedMissionButton = null;

_retreatButton
```

```
= null;

if (_settingsButton)
_settingsButton.destroy();

_settingsButton = null;

if (_switchStateButton)
_switchStateButton.destroy();

_switchStateButton = null;
}
}
}

-----------------------------------
File 743: igw\com\ui\hud\shared\PlayerView.as
package com.ui.hud.shared
{
import com.Application;
import com.enum.PlayerUpdateEnum;
import com.enum.PositionEnum;
import com.enum.ToastEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.StateEvent;
import com.event.signal.TransactionSignal;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.starbase.BuffVO;
import com.model.transaction.TransactionVO;
import com.presenter.shared.IUIPresenter;
import com.service.ExternalInterfaceAPI;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.core.effects.EffectFactory;
import com.ui.hud.shared.engineering.BuffButton;
import com.ui.modal.playerinfo.PlayerProfileView;
import com.ui.modal.store.StoreView;
import com.util.CommonFunctionUtil;

import
```

```
flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.Timer;

import org.adobe.utils.StringUtil;
import org.parade.enum.PlatformEnum;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class PlayerView extends View
{
private var _addMoreBtn:BitmapButton;
private var _avatar:ImageComponent;
private var _avatarFrame:ScaleBitmap;
private var _bg:ScaleBitmap;
private var _buffIcons:Vector.<BuffButton>;
private var _buffLookup:Dictionary;
private var _levelLbl:Label;
private var _nameLbl:Label;
private var _palladiumBG:Sprite;
private var _palladiumSymbol:Bitmap;
private var _premiumLbl:Label;
private var _starbaseRating:Label;
private var _storeBtn:BitmapButton;
private var _timer:Timer;
private var _tooltip:Tooltips;
private var _xpBar:ProgressBar;

private var _store:String = 'CodeString.Controls.Store'; //STORE
private var _level:String = 'CodeString.UserView.Level'; //<FONT COLOR="#d1e5f7">LEVEL
</FONT><FONT COLOR="#ecffff"><b>[[Number.Level]]</b></FONT>
private var _starbaseRatingText:String = 'CodeString.UserView.StarbaseRating'; //<FONT
COLOR="#d1e5f7">STARBASE RATING </FONT><FONT
COLOR="#ecffff"><b>[[Number.Level]]</b></FONT>
private var _palladiumText:String = 'CodeString.Shared.Palladium'; //PALLADIUM
private var _xpTooltipStr:String = 'CodeString.UserView.XpTooltip'; //Experience Points:
<br>Current: [[Number.CurrentXp]]<br>Next Lvl: [[Number.NextLevelXp]]
private var _palladiumTooltipText:String = 'CodeString.UserView.GetMorePaladiumTooltip';
//Get More Palladium
private var _playerProfileTooltipText:String = 'CodeString.UserView.PlayerProfileTooltip';
//Player Profile

[PostConstruct]
override public function init():void
{
```

```
super.init();
presenter.addTransactionListener(TransactionSignal.DATA_IMPORTED, onBuffsImported);

_buffIcons = new Vector.<BuffButton>;
_buffLookup = new Dictionary();

_bg = UIFactory.getPanel(PanelEnum.PLAYER_CONTAINER_NOTCHED, 367, 136);

_avatarFrame = UIFactory.getPanel(PanelEnum.CHARACTER_FRAME, 125, 125, 6, 5);
_avatar = ObjectPool.get(ImageComponent);
_avatar.init(125, 125);
_avatar.x = 7;
_avatar.y = 5;
_avatar.center = true;
_avatar.buttonMode = true;
presenter.loadPortraitMedium(CurrentUser.avatarName, _avatar.onImageLoaded);

_nameLbl = UIFactory.getLabel(LabelEnum.H4, 190, 33, 132);
_nameLbl.textColor = 0xecffff;
_nameLbl.useLocalization = false;
_nameLbl.align = TextFormatAlign.LEFT;
_nameLbl.text = CurrentUser.name;

_levelLbl = UIFactory.getLabel(LabelEnum.H4, 132, 19, 132, 19);
_levelLbl.align = TextFormatAlign.LEFT;
_levelLbl.bold = false;
_levelLbl.setHtmlTextWithTokens(_level, {'[[Number.Level]]':CurrentUser.level});

_xpBar = UIFactory.getProgressBar(UIFactory.getPanel(PanelEnum.STATBAR, 217, 21),
UIFactory.getPanel(PanelEnum.STATBAR_CONTAINER, 226, 29), 0, 1, 0, 134, 44);

_storeBtn = UIFactory.getButton(ButtonEnum.GOLD_A, 126, 53, 235, 77, _store,
LabelEnum.H3);
_storeBtn.label.y += 4;

_palladiumBG =
UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL,
PanelEnum.HEADER_NOTCHED, 96, 35, 18, 135, 77, _palladiumText, LabelEnum.H5);
_palladiumSymbol = UIFactory.getBitmap('IconPalladiumBMD');
_palladiumSymbol.x = 6;
_palladiumSymbol.y = 21;

_premiumLbl = new Label(16, 0xf0f0f0, 41, 25);
_premiumLbl.constrictTextToSize = false;
_premiumLbl.align = TextFormatAlign.RIGHT;
_premiumLbl.text = String(CurrentUser.wallet.premium);
_premiumLbl.x = 26;
_premiumLbl.y = 26;
_premiumLbl.mouseEnabled = true;

_addMoreBtn
```

```
= UIFactory.getButton(ButtonEnum.PLUS, 0, 0, 74, 28);
_addMoreBtn.hitArea = _palladiumBG;

_palladiumBG.addChild(_palladiumSymbol);
_palladiumBG.addChild(_premiumLbl);
_palladiumBG.addChild(_addMoreBtn);

_starbaseRating = UIFactory.getLabel(LabelEnum.H4, 145, 25, 367 - 155, _levelLbl.y);
_starbaseRating.align = TextFormatAlign.RIGHT;
_starbaseRating.bold = false;
_starbaseRating.setHtmlTextWithTokens(_starbaseRatingText,
{'[[Number.Level]]':CurrentUser.baseRating});

//transaction update timer
_timer = new Timer(1000);
addListener(_timer, TimerEvent.TIMER, onTimer);
_timer.start();

var loc:Localization = Localization.instance;
_tooltip.addTooltip(_avatar, this, null, loc.getString(_playerProfileTooltipText));
_tooltip.addTooltip(_palladiumBG, this, null, loc.getString(_palladiumTooltipText));
_tooltip.addTooltip(_xpBar, this, getXpTooltip, '', 250, 180, 18, true);

addListener(_addMoreBtn, MouseEvent.CLICK, onAddMoreClick);
addListener(_avatar, MouseEvent.CLICK, onPortraitClick);
addListener(_storeBtn, MouseEvent.CLICK, onStoreClick);

addChild(_bg);
addChild(_avatar);
addChild(_avatarFrame);
addChild(_storeBtn);
addChild(_palladiumBG);
addChild(_nameLbl);
addChild(_levelLbl);
addChild(_starbaseRating);
addChild(_xpBar);

x = y = 4;

addEffects();
effectsIN();

CurrentUser.onPlayerUpdate.add(onPlayerUpdated);
CurrentUser.wallet.onPremiumChange.add(updatePremiumUI);
addHitArea();
updateXP(0, CurrentUser.xp);
onBuffsImported();
onStageResize();

visible = !presenter.inFTE;
}
```

```
private function loadIcon( prototype:IPrototype, callback:Function ):void
{
var icon:String = presenter.getPrototypeUIIcon(prototype);
presenter.loadIcon(icon, callback);
}

private function onPlayerUpdated( updateType:int, oldValue:String, newValue:String ):void
{
if (oldValue != newValue)
{
switch (updateType)
{
case PlayerUpdateEnum.TYPE_XP:
updateXP(int(oldValue), int(newValue));
break;
case PlayerUpdateEnum.TYPE_NAME:
if (_nameLbl)
_nameLbl.text = newValue;
break;
case PlayerUpdateEnum.TYPE_BASERATING:
if (_starbaseRating)
_starbaseRating.setHtmlTextWithTokens(_starbaseRatingText,
{'[[Number.Level]]':CurrentUser.baseRating});

ExternalInterfaceAPI.shareLevelUp(CurrentUser.name, CurrentUser.baseRating, true);
break;
}
}
}

private function updateXP( oldValue:int, newValue:int ):void
{
var currentXP:int = CurrentUser.xp;
var nextLevelMinExp:int =
PrototypeModel.instance.getConstantPrototypeValueByName(CommonFunctionUtil.getLevelProtoName(C
+ 1));
if (nextLevelMinExp <= currentXP)
{
CurrentUser.level = CommonFunctionUtil.findPlayerLevel(currentXP);
if (_levelLbl)
_levelLbl.setHtmlTextWithTokens(_level, {'[[Number.Level]]':CurrentUser.level});
if (CurrentUser.level > 1 && oldValue != 0 && oldValue != newValue)
{
showToast(ToastEnum.LEVEL_UP);
if (DeviceMetrics.PLATFORM == PlatformEnum.BROWSER)
{
ExternalInterfaceAPI.shareLevelUp(CurrentUser.name, CurrentUser.level, false);
}
if
```

```
(Application.STATE == StateEvent.GAME_STARBASE)
presenter.updateStarbasePlatform();
}
}
currentXP =
PrototypeModel.instance.getConstantPrototypeValueByName(CommonFunctionUtil.getLevelProtoName(C
nextLevelMinExp =
PrototypeModel.instance.getConstantPrototypeValueByName(CommonFunctionUtil.getLevelProtoName(C
+ 1));
if (_xpBar)
_xpBar.amount = (CurrentUser.xp - currentXP) / (nextLevelMinExp - currentXP);
}

private function onBuffsImported( data:TransactionVO = null ):void
{
var buff:BuffVO;
var buffButton:BuffButton;

//base bubble
if (presenter.bubbleTimeRemaining > 0 && !_buffLookup.hasOwnProperty("Protection"))
{
buffButton = ObjectPool.get(BuffButton);
buffButton.init(null, presenter);
_buffIcons.push(buffButton);
_buffLookup['Protection'] = buffButton;
addChild(buffButton);
layoutBuffs();
addListener(buffButton, MouseEvent.CLICK, onBuffClicked);
_tooltip.addTooltip(buffButton, this, buffButton.getTooltip);
}

//normal buffs
var buffs:Vector.<BuffVO> = presenter.buffs;
for (var i:int = 0; i < buffs.length; i++)
{
buff = buffs[i];
if (!_buffLookup.hasOwnProperty(buff.buffType))
{
buffButton = ObjectPool.get(BuffButton);
buffButton.init(buff, presenter);
_buffIcons.push(buffButton);
_buffLookup[buff.buffType] = buffButton;
addChild(buffButton);
layoutBuffs();
addListener(buffButton, MouseEvent.CLICK, onBuffClicked);
_tooltip.addTooltip(buffButton, this, buffButton.getTooltip);
}
}
}

private
```

```
function layoutBuffs():void
{
var buffButton:BuffButton;
for (var i:int = 0; i < _buffIcons.length; i++)
{
buffButton = _buffIcons[i];
buffButton.x = _bg.width - 30 - ((i % 6) * 34);
buffButton.y = _bg.height + 4 + (Math.floor(i / 6) * 54);
}
}

private function onBuffClicked( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
var storeView:StoreView = StoreView(showView(StoreView));
if (e.currentTarget is BuffButton)
{
var button:BuffButton = BuffButton(e.currentTarget);
if (button.buff)
storeView.openToBuffsAndFilter(StoreView.FILTER_ALL);
else
storeView.openToProtectionAndFilter(StoreView.FILTER_ALL);
} else
storeView.openToBuffsAndFilter(StoreView.FILTER_ALL);
}

private function onTimer( e:TimerEvent ):void
{
//update buff timers
var buffButton:BuffButton;
for (var i:int = 0; i < _buffIcons.length; i++)
{
if (!_buffIcons[i].updateTime())
{
//time expired. remove buff
buffButton = _buffIcons[i];
delete _buffLookup[buffButton.buffType];
if (buffButton.buff != null)
presenter.removeBuff(buffButton.buff);
removeListener(buffButton, MouseEvent.CLICK, onBuffClicked);
_tooltip.removeTooltip(buffButton);
ObjectPool.give(buffButton);
_buffIcons.splice(i, 1);
i--;
layoutBuffs();
}
}
}

override
```

```
protected function onStateChange( state:String ):void
{
switch (state)
{
case StateEvent.GAME_BATTLE:
destroy();
break;
}
}

private function onPortraitClick( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
var playerProfileView:PlayerProfileView =
PlayerProfileView(_viewFactory.createView(PlayerProfileView));
playerProfileView.playerKey = CurrentUser.id;
_viewFactory.notify(playerProfileView);
e.stopPropagation();
}

private function onAddMoreClick( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;

CommonFunctionUtil.popPaywall();

//if (Application.NETWORK != Application.NETWORK_KONGREGATE)
//{
//CommonFunctionUtil.popPaywall();
//}
//else
//{
//_viewFactory.openPayment();
//}

e.stopPropagation();
}

private function updatePremiumUI( isAdded:Boolean ):void
{
if (_premiumLbl)
_premiumLbl.text = String(CurrentUser.wallet.premium);

if (isAdded)
showToast(ToastEnum.PALLADIUM_ADDED);
}

private function premiumUITooltip():String
{
```

```
return String(CurrentUser.wallet.premium);
}

private function onStoreClick( e:Event ):void
{
if (!presenter.hudEnabled)
return;
if (presenter.hudEnabled)
showView(StoreView);
e.stopPropagation();
}

private function getXpTooltip():String
{
var currentXP:Number = CurrentUser.xp;
var nextLevelMinExp:Number =
presenter.getConstantPrototypeValueByName(CommonFunctionUtil.getLevelProtoName(CurrentUser.leve
+ 1));

var xpTooltipDict:Dictionary = new Dictionary();
xpTooltipDict['[[Number.CurrentXp]]'] = StringUtil.commaFormatNumber(currentXP);
xpTooltipDict['[[Number.NextLevelXp]]'] = StringUtil.commaFormatNumber(nextLevelMinExp);

return Localization.instance.getStringWithTokens(_xpTooltipStr, xpTooltipDict);
}

override protected function addEffects():void {
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.TOP,
onStageResize)); }

private function onStageResize( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
}

public function get premiumBg():Sprite { return _palladiumBG; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function get screenshotBlocker():Boolean {return true;}

[Inject]
public function set tooltip( v:Tooltips ):void { _tooltip = v; }

override public function get type():String { return ViewEnum.UI; }

override public function destroy():void
{
```

```
CurrentUser.onPlayerUpdate.remove(onPlayerUpdated);
presenter.removeTransactionListener(onBuffsImported);
CurrentUser.wallet.onPremiumChange.remove(updatePremiumUI);
super.destroy();

_addMoreBtn = UIFactory.destroyButton(_addMoreBtn);
_avatar.destroy();
_avatar = null;
_avatarFrame = UIFactory.destroyPanel(_avatarFrame);
_bg = UIFactory.destroyPanel(_bg);

//destroy the buffs
for (var i:int = 0; i < _buffIcons.length; i++)
ObjectPool.give(_buffIcons[i]);
_buffIcons.length = 0;
_buffIcons = null;
_buffLookup = null;

_levelLbl = UIFactory.destroyLabel(_levelLbl);
_nameLbl = UIFactory.destroyLabel(_nameLbl);
_palladiumBG = null;
_palladiumSymbol = null;
_premiumLbl = UIFactory.destroyLabel(_premiumLbl);
_storeBtn = UIFactory.destroyButton(_storeBtn);

_timer.reset();
_timer = null;

_tooltip.removeTooltip(null, this);
_tooltip = null;

_xpBar.destroy();
_xpBar = null;
}
}
}

------------------------------------
File 744: igw\com\ui\hud\shared\bridge\AchievementRiverButton.as
package com.ui.hud.shared.bridge
{

import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.Sprite;
import
```

```
flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;

public class AchievementRiverButton extends Sprite
{
public var onClick:Signal;

private var _achievementIcon:Bitmap;
private var _achievementBtn:BitmapButton;
private var _btnText:Label;

private var _achievementText:String = 'CodeString.Achievements.RiverBtn'; //BADGES

public function AchievementRiverButton()
{
onClick = new Signal();

_achievementIcon = UIFactory.getBitmap('IconTrophyBMD');

_achievementBtn = UIFactory.getButton(ButtonEnum.ICON_FRAME, 60, 60);
_achievementBtn.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);

_btnText = new Label(20, 0xd1e5f7, 100, 25);
_btnText.bold = true;
_btnText.constrictTextToSize = false;
_btnText.align = TextFormatAlign.CENTER;
_btnText.text = _achievementText;

addChild(_achievementBtn);
addChild(_achievementIcon);
addChild(_btnText);

layout();
}

private function onMouseClick( e:MouseEvent ):void
{
if (onClick)
onClick.dispatch();
e.stopPropagation();
}

private function layout():void
{
_achievementIcon.x = _achievementBtn.x + (_achievementBtn.defaultSkinWidth -
_achievementIcon.width) * 0.5;
_achievementIcon.y = _achievementBtn.y + (_achievementBtn.defaultSkinHeight -
_achievementIcon.height) * 0.5;

_btnText.x
```

```
= _achievementBtn.x + (_achievementBtn.width - _btnText.width) * 0.5;
_btnText.y = _achievementBtn.height - 1;
}

public function destroy():void
{
if (onClick)
onClick.removeAll();

onClick = null;

if (_achievementBtn)
{
_achievementBtn.removeEventListener(MouseEvent.CLICK, onMouseClick);
_achievementBtn.destroy();
}

_achievementBtn = null;

if (_btnText)
_btnText.destroy();

_btnText = null;

_achievementIcon = null;
}

}
}

-------------------------------------
File 745: igw\com\ui\hud\shared\bridge\BridgeView.as
package com.ui.hud.shared.bridge
{
import com.Application;
import com.enum.PositionEnum;
import com.event.StateEvent;
import com.model.event.EventVO;
import com.model.player.CurrentUser;
import com.model.player.OfferVO;
import com.presenter.shared.IUIPresenter;
import com.ui.core.View;
import com.ui.core.effects.EffectFactory;
import com.ui.modal.achievements.AchievementView;
import com.ui.modal.event.EventView;
import com.ui.modal.information.MessageOfTheDayView;
import com.ui.modal.intro.FAQView;
import com.ui.modal.offers.OfferView;

import flash.events.Event;
import
```

```
flash.geom.Rectangle;

import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;
//import flash.external.ExternalInterface;

public class BridgeView extends View
{
private var _offerBtn:OfferRiverButton;
private var _missionRiver:MissionRiver;
private var _achievementRiverBtn:AchievementRiverButton;
private var _eventRiverBtn:EventRiverButton;
private var _faqRiverBtn:FAQRiverButton;
//private var _discordBtn:DiscordButton;

private var _offers:Vector.<OfferVO>;

[PostConstruct]
override public function init():void
{
super.init();
var bgRect:Rectangle = new Rectangle(14, 133, 40, 2);
x = 3;
y = 155;
_offers = CurrentUser.offers;
CurrentUser.onPlayerOffersUpdated.add(onOffersUpdated);
presenter.addEventUpdatedListener(onEventUpdated);

setUpBridge();
addHitArea();
addEffects();
effectsIN();
onStageResized();

visible = !presenter.inFTE;
}

private function setUpBridge():void
{
_offerBtn = ObjectPool.get(OfferRiverButton);
_offerBtn.offers = _offers;
_offerBtn.onClick.add(onOfferClicked);
addChild(_offerBtn);

//mission river
_missionRiver = ObjectPool.get(MissionRiver);
presenter.injectObject(_missionRiver);
addChild(_missionRiver);

_achievementRiverBtn
```

```
= ObjectPool.get(AchievementRiverButton);
_achievementRiverBtn.onClick.add(onAchievementRiverClick);
addChild(_achievementRiverBtn);

_eventRiverBtn = ObjectPool.get(EventRiverButton);
_eventRiverBtn.updatedEvents(presenter.currentActiveEvent, presenter.activeEvents,
presenter.upcomingEvents);
_eventRiverBtn.onClick.add(onEventRiverClick);
_eventRiverBtn.onUpdated.add(layout);
addChild(_eventRiverBtn);

_faqRiverBtn = ObjectPool.get(FAQRiverButton);
_faqRiverBtn.onClick.add(onFAQRiverClick);
addChild(_faqRiverBtn);

//_discordBtn = ObjectPool.get(DiscordButton);
//_discordBtn.onClick.add(onDiscordClick);
//addChild(_discordBtn);

layout();
}

private function onEventUpdated( currentActiveEvent:EventVO,
activeEvents:Vector.<EventVO>, upcomingEvents:Vector.<EventVO> ):void
{
if (_eventRiverBtn)
_eventRiverBtn.updatedEvents(currentActiveEvent, activeEvents, upcomingEvents);
}

private function onOffersUpdated( v:Vector.<OfferVO> ):void
{
if (_offerBtn)
_offerBtn.offers = v;
}

private function onOfferClicked( currentOffer:OfferVO ):void
{
if (currentOffer && presenter.hudEnabled)
{
var offerView:OfferView = OfferView(_viewFactory.createView(OfferView));
offerView.offerProtoName = currentOffer;
_viewFactory.notify(offerView);
}
}

private function onEventRiverClick():void
{
var view:*;
if (_eventRiverBtn.hasScore())
{
view
```

```
= EventView(_viewFactory.createView(EventView));
} else
{
view = MessageOfTheDayView(_viewFactory.createView(MessageOfTheDayView));
}

_viewFactory.notify(view);
}

private function onAchievementRiverClick():void
{
if (!presenter.hudEnabled)
return;
var achievementView:AchievementView =
AchievementView(_viewFactory.createView(AchievementView));
_viewFactory.notify(achievementView);
}

private function onFAQRiverClick():void
{
if (!presenter.hudEnabled)
return;
var faqView:FAQView = FAQView(_viewFactory.createView(FAQView));
_viewFactory.notify(faqView);
}

// private function onDiscordClick():void
// {
// if (!presenter.hudEnabled)
// return;
// ExternalInterface.call("window.open", "https://discord.gg/igw", "_blank"); //Open Discord
// }

override protected function onStateChange( state:String ):void
{
if (state == StateEvent.GAME_BATTLE)
destroy();
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.CENTER,
onStageResized));
}

private function onStageResized( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
x = 3;
y = 166 * Application.SCALE;
layout();
```

```
}

private function layout():void
{
var xPos:uint = 12;
var yPos:uint = 15;

var height:Number = 0;

if (_offerBtn)
{
_offerBtn.x = xPos - 22;
_offerBtn.y = yPos;
yPos += _offerBtn.height + 4;
}

//to not break the fte we can't do a visibility check
if (_missionRiver)
{
_missionRiver.x = xPos;
_missionRiver.y = yPos;
yPos += _missionRiver.height - 3;
}

if (_achievementRiverBtn)
{
_achievementRiverBtn.x = xPos;
_achievementRiverBtn.y = yPos;
yPos += _achievementRiverBtn.height + 3;
}

if (_eventRiverBtn && _eventRiverBtn.visible)
{
_eventRiverBtn.x = xPos;
_eventRiverBtn.y = yPos;
yPos += _eventRiverBtn.height + 3;
}

if (_faqRiverBtn)
{
if (DeviceMetrics.HEIGHT_PIXELS < 890 && (_eventRiverBtn && _eventRiverBtn.visible))
{
_faqRiverBtn.x = _offerBtn.x + _offerBtn.width - 50;
_faqRiverBtn.y = _eventRiverBtn.y;
} else
{
_faqRiverBtn.x = xPos;
_faqRiverBtn.y = yPos;
}
}
```

```
// if (_discordBtn)
// {
// if (DeviceMetrics.HEIGHT_PIXELS < 890 && (_eventRiverBtn && _eventRiverBtn.visible))
// {
// _discordBtn.x = _offerBtn.x + _offerBtn.width - 50;
// _discordBtn.y = _achievementRiverBtn.y;
// } else
// {
// yPos += _faqRiverBtn.height + 3;
// _discordBtn.x = xPos;
// _discordBtn.y = yPos;

// }
// }
}

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function get type():String { return ViewEnum.UI }

override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
CurrentUser.onPlayerOffersUpdated.remove(onOffersUpdated);
presenter.removeEventUpdatedListener(onEventUpdated);

//Offer River
if (_offerBtn)
ObjectPool.give(_offerBtn);

_offerBtn = null;

//Mission river
if (_missionRiver)
ObjectPool.give(_missionRiver);

_missionRiver = null;

//Achievement River
if (_achievementRiverBtn)
ObjectPool.give(_achievementRiverBtn);

_achievementRiverBtn = null;

//Event River
if
```

```
(_eventRiverBtn)
ObjectPool.give(_eventRiverBtn);

_eventRiverBtn = null;

//FAQ River
if (_faqRiverBtn)
ObjectPool.give(_faqRiverBtn);

_faqRiverBtn = null;

//Discord Button
// if (_discordBtn)
// ObjectPool.give(_discordBtn);

// _discordBtn = null;


super.destroy();
}
}
}

-----------------------------------
File 746: igw\com\ui\hud\shared\bridge\DiscordButton.as
package com.ui.hud.shared.bridge
{
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;

public class DiscordButton extends Sprite
{
public var onClick:Signal;

private var _discordIcon:Bitmap;
private var _discordBtn:BitmapButton;
private var _btnText:Label;

private var _helpText:String = 'CodeString.Discord.Link'; //Discord

public
```

```
function DiscordButton()
{
onClick = new Signal();

var discordIconClass:Class = Class(getDefinitionByName('DiscordIconBMD')); //Discord Icon
_discordIcon = new Bitmap(BitmapData(new discordIconClass()));

_discordBtn = UIFactory.getButton(ButtonEnum.ICON_FRAME, 60, 60);
_discordBtn.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);

_btnText = new Label(20, 0xd1e5f7, 100, 25);
_btnText.bold = true;
_btnText.constrictTextToSize = false;
_btnText.align = TextFormatAlign.CENTER;
_btnText.text = _helpText;

addChild(_discordBtn);
addChild(_discordIcon);
addChild(_btnText);

layout();
}

private function onMouseClick( e:MouseEvent ):void
{
if (onClick)
onClick.dispatch();
e.stopPropagation();
}

private function layout():void
{
_discordIcon.x = _discordBtn.x + (_discordBtn.defaultSkinWidth - _discordIcon.width) * 0.5;
_discordIcon.y = _discordBtn.y + (_discordBtn.defaultSkinHeight - _discordIcon.height) * 0.5;

_btnText.x = _discordBtn.x + (_discordBtn.width - _btnText.width) * 0.5;
_btnText.y = _discordBtn.height - 1;
}

public function destroy():void
{
if (onClick)
onClick.removeAll();

onClick = null;

if (_discordBtn)
{
_discordBtn.removeEventListener(MouseEvent.CLICK, onMouseClick);
_discordBtn.destroy();
}
```

```
_discordBtn = null;

if (_btnText)
_btnText.destroy();

_btnText = null;

_discordIcon = null;
}

}
}

-----------------------------------
File 747: igw\com\ui\hud\shared\bridge\EventRiverButton.as
package com.ui.hud.shared.bridge
{
import com.enum.ui.ButtonEnum;
import com.model.event.EventVO;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Timer;

import org.osflash.signals.Signal;

public class EventRiverButton extends Sprite
{
public var onClick:Signal;
public var onUpdated:Signal;

private var _eventIcon:Bitmap;
private var _eventBtn:BitmapButton;
private var _btnText:Label;

private var _eventTimeRemaining:Label;

private var _timer:Timer;

private var _currentEvent:EventVO;

private
```

```
var _eventText:String = 'CodeString.EventRiver.IncursionEvent'; //INCURSION
private var _eventUpcomingText:String = 'CodeString.EventRiver.StartIncursionEvent';
//INCURSION STARTS

public function EventRiverButton()
{
onClick = new Signal();
onUpdated = new Signal();

_eventIcon = UIFactory.getBitmap('IconEventBMD');

_eventBtn = UIFactory.getButton(ButtonEnum.ICON_FRAME, 60, 60);
_eventBtn.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);

_btnText = new Label(20, 0xd1e5f7, 100);
_btnText.constrictTextToSize = false;
_btnText.autoSize = TextFieldAutoSize.CENTER;
_btnText.align = TextFormatAlign.CENTER;
_btnText.multiline = true;
_btnText.text = _eventText;

_eventTimeRemaining = new Label(14, 0xd1e5f7, 150, 25);
_eventTimeRemaining.constrictTextToSize = false;
_eventTimeRemaining.align = TextFormatAlign.CENTER;

_timer = new Timer(1000);
_timer.addEventListener(TimerEvent.TIMER, onTimerTick, false, 0, true);

addChild(_eventBtn);
addChild(_eventIcon);
addChild(_btnText);
addChild(_eventTimeRemaining);

layout();
}

public function updatedEvents( currentActiveEvent:EventVO, activeEvents:Vector.<EventVO>,
upcomingEvents:Vector.<EventVO> ):void
{
var active:Boolean;
_currentEvent = currentActiveEvent;

if (_currentEvent)
{
active = (activeEvents.indexOf(_currentEvent) != -1);

_btnText.text = (active) ? _eventText : _eventUpcomingText;
visible = true;
_timer.start();

onUpdated.dispatch();
```

```
} else
{
_btnText.text = _eventText;
visible = false;
if (_timer)
_timer.stop();

onUpdated.dispatch();
}

layout();
}

private function onTimerTick( e:TimerEvent ):void
{
if (_currentEvent == null)
{
if (_timer)
_timer.stop();

return;
}
var timeRemaining:Number = _currentEvent.timeRemainingMS;

if (timeRemaining <= 0)
{
_timer.stop();
_currentEvent = null;
} else
{
if (_eventTimeRemaining)
_eventTimeRemaining.setBuildTime(timeRemaining / 1000);
}
}

private function onMouseClick( e:MouseEvent ):void
{
if (onClick)
onClick.dispatch();
e.stopPropagation();
}

public function uiTracked():Boolean
{
if (_currentEvent)
return _currentEvent.isUiTracking;

return false;
}
```

```
public function hasScore():Boolean
{
if (_currentEvent)
return _currentEvent.hasScore;

return false;
}

private function layout():void
{
_eventIcon.x = _eventBtn.x + (_eventBtn.defaultSkinWidth - _eventIcon.width) * 0.5;
_eventIcon.y = _eventBtn.y + (_eventBtn.defaultSkinHeight - _eventIcon.height) * 0.5;

_btnText.x = _eventBtn.x + (_eventBtn.width - _btnText.width) * 0.5;
_btnText.y = _eventBtn.height - 1;

_eventTimeRemaining.x = _eventBtn.x + (_eventBtn.width - _eventTimeRemaining.width) * 0.5;
_eventTimeRemaining.y = _btnText.y + _btnText.textHeight;
}

override public function get height():Number
{
if (_eventTimeRemaining.visible)
return (_eventTimeRemaining.y + _eventTimeRemaining.height);
else
return (_btnText.y + _btnText.height);
}

public function destroy():void
{
if (onClick)
onClick.removeAll();

onClick = null;

if (onUpdated)
onUpdated.removeAll();

onUpdated = null;

if (_eventBtn)
{
_eventBtn.removeEventListener(MouseEvent.CLICK, onMouseClick);
_eventBtn.destroy();
}

_eventBtn = null;

if
```

```
(_timer)
{
_timer.removeEventListener(TimerEvent.TIMER, onTimerTick);
if (_timer.running)
_timer.stop();
}
_timer = null;

if (_eventTimeRemaining)
_eventTimeRemaining.destroy();

_eventTimeRemaining = null;

if (_btnText)
_btnText.destroy();

_btnText = null;

_eventIcon = null;
}
}
}


-------------------------------------
File 748: igw\com\ui\hud\shared\bridge\FAQRiverButton.as
package com.ui.hud.shared.bridge
{
import com.enum.ui.ButtonEnum;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;

public class FAQRiverButton extends Sprite
{
public var onClick:Signal;

private var _faqIcon:Bitmap;
private var _faqBtn:BitmapButton;
private var _btnText:Label;

private
```

```
var _helpText:String = 'CodeString.FAQRiver.Help'; //HELP

public function FAQRiverButton()
{
onClick = new Signal();

var faqIconClass:Class = Class(getDefinitionByName('FAQEightsIconBMD'));
_faqIcon = new Bitmap(BitmapData(new faqIconClass()));

_faqBtn = UIFactory.getButton(ButtonEnum.ICON_FRAME, 60, 60);
_faqBtn.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);

_btnText = new Label(20, 0xd1e5f7, 100, 25);
_btnText.bold = true;
_btnText.constrictTextToSize = false;
_btnText.align = TextFormatAlign.CENTER;
_btnText.text = _helpText;

addChild(_faqBtn);
addChild(_faqIcon);
addChild(_btnText);

layout();
}

private function onMouseClick( e:MouseEvent ):void
{
if (onClick)
onClick.dispatch();
e.stopPropagation();
}

private function layout():void
{
_faqIcon.x = _faqBtn.x + (_faqBtn.defaultSkinWidth - _faqIcon.width) * 0.5;
_faqIcon.y = _faqBtn.y + (_faqBtn.defaultSkinHeight - _faqIcon.height) * 0.5;

_btnText.x = _faqBtn.x + (_faqBtn.width - _btnText.width) * 0.5;
_btnText.y = _faqBtn.height - 1;
}

public function destroy():void
{
if (onClick)
onClick.removeAll();

onClick = null;

if (_faqBtn)
{
_faqBtn.removeEventListener(MouseEvent.CLICK,
```

```
onMouseClick);
_faqBtn.destroy();
}

_faqBtn = null;

if (_btnText)
_btnText.destroy();

_btnText = null;

_faqIcon = null;
}

}
}

------------------------------------
File 749: igw\com\ui\hud\shared\bridge\MissionRiver.as
package com.ui.hud.shared.bridge
{
import com.enum.ToastEnum;
import com.event.MissionEvent;
import com.event.ToastEvent;
import com.model.mission.MissionInfoVO;
import com.model.mission.MissionVO;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.IMissionPresenter;
import com.ui.modal.mission.captainslog.CaptainsLogView;

import flash.display.Sprite;
import flash.events.MouseEvent;

import org.parade.core.IViewFactory;
import org.shared.ObjectPool;

public class MissionRiver extends Sprite
{
private var _initialized:Boolean;
private var _missionButton:MissionRiverButton;
private var _missionName:String;
private var _presenter:IMissionPresenter;
private var _viewFactory:IViewFactory;

private var _completeImage:String = "complete.jpg";

[PostConstruct]
public function init():void
{
_initialized = false;
_presenter
```

```
&& _presenter.highfive();
presenter.addTransactionListener(onMissionUpdated);

//create the mission button
_missionButton = new MissionRiverButton();
_missionButton.addEventListener(MouseEvent.CLICK, onMissionClicked, false, 0, true);
addChild(_missionButton);

onMissionUpdated(null);
}

private function onMissionClicked( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
var mission:MissionVO = presenter.currentMission;
if (!mission.accepted)
{
presenter.dispatchMissionEvent(MissionEvent.MISSION_GREETING);
} else if (mission.complete && !mission.rewardAccepted)
{
presenter.dispatchMissionEvent(MissionEvent.MISSION_VICTORY);
} else
{
var log:CaptainsLogView = CaptainsLogView(_viewFactory.createView(CaptainsLogView));
_viewFactory.notify(log);
}
e.stopPropagation();
}

/**
* Called when the server updates a mission.
* Updates the missionButton and if this is a new mission the button image is changed.
*/
private function onMissionUpdated( transaction:TransactionVO ):void
{
var mission:MissionVO = presenter.currentMission;
if (_missionName != mission.name)
{
_missionButton.mission = mission;
if (!mission.isFTE)
{
var info:MissionInfoVO = presenter.getMissionInfo(MissionEvent.MISSION_GREETING);
var icon:String;
if (mission.accepted && mission.complete && mission.rewardAccepted)
icon = _completeImage;
else
icon = info.smallImage;

presenter.loadIcon(icon, _missionButton.onImageLoaded);
_missionName
```

```
= mission.name;
ObjectPool.give(info);
if (!mission.accepted && _initialized)
{
//send out a toast to inform the player they have a new mission
var toastEvent:ToastEvent = new ToastEvent();
toastEvent.toastType = ToastEnum.MISSION_NEW;
presenter.dispatch(toastEvent);
}
_initialized = visible = true;
}
} else
_missionButton.mission = mission;
}

[Inject]
public function set presenter( v:IMissionPresenter ):void { _presenter = v; }
public function get presenter():IMissionPresenter { return IMissionPresenter(_presenter); }

[Inject]
public function set viewFactory( v:IViewFactory ):void { _viewFactory = v; }

public function destroy():void
{
presenter.removeTransactionListener(onMissionUpdated);

_missionButton.removeEventListener(MouseEvent.CLICK, onMissionClicked);
_missionButton.destroy();
_missionButton = null;
_presenter && _presenter.shun();
_presenter = null;
_viewFactory = null;
}
}
}



---
File 750: igw\com\ui\hud\shared\bridge\MissionRiverButton.as
package com.ui.hud.shared.bridge
{
import com.enum.ui.ButtonEnum;
import com.model.mission.MissionVO;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Bitmap;
import
```

```
flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class MissionRiverButton extends Sprite
{
private var _exclamation:Bitmap;
private var _image:ImageComponent;
private var _mission:MissionVO;
private var _question:Bitmap;

public var onClick:Signal;

private var _faqBtn:BitmapButton;
private var _btnText:Label;

private var _missionText:String = 'CodeString.MissionRiver.Mission'; //MISSION

public function MissionRiverButton()
{
onClick = new Signal();

_image = ObjectPool.get(ImageComponent);
_image.mouseEnabled = false;
_image.init(60, 60);
_image.center = true;

_faqBtn = UIFactory.getButton(ButtonEnum.ICON_FRAME, 60, 60);
_faqBtn.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);

_btnText = new Label(20, 0xd1e5f7, 100, 25);
_btnText.bold = true;
_btnText.constrictTextToSize = false;
_btnText.align = TextFormatAlign.CENTER;
_btnText.text = _missionText;

_exclamation = UIFactory.getBitmap('IconMissionExclamationBMD');
_exclamation.x = 60 - _exclamation.width + 7;
_exclamation.y = -_exclamation.height * .15;
_exclamation.visible = false;

_question = UIFactory.getBitmap('IconMissionQuestionBMD');
_question.x = 60 - _question.width + 7;
_question.y = -_question.height * .15;
_question.visible
```

```
= false;

addChild(_faqBtn);
addChild(_image);
addChild(_exclamation);
addChild(_question);
addChild(_btnText);

layout();
}

private function onMouseClick( e:MouseEvent ):void
{
if (onClick)
onClick.dispatch();
}

private function layout():void
{
if (_btnText)
{
_btnText.x = _faqBtn.x + (_faqBtn.width - _btnText.width) * 0.5;
_btnText.y = _faqBtn.height - 1;
}
}

public function onImageLoaded( asset:BitmapData ):void
{
if (_image)
{
_image.onImageLoaded(asset);
layout();
}
}

private function onFadeOut():void
{
if (_exclamation.visible)
TweenLite.to(_exclamation, .5, {alpha:1.0, ease:Quad.easeOut, onComplete:onFadeIn});
else if (_question.visible)
TweenLite.to(_question, .5, {alpha:1.0, ease:Quad.easeOut, onComplete:onFadeIn});
}

private function onFadeIn():void
{
if (_exclamation.visible)
TweenLite.to(_exclamation, .5, {alpha:0.1, ease:Quad.easeIn, onComplete:onFadeOut});
else if (_question.visible)
TweenLite.to(_question, .5, {alpha:0.1, ease:Quad.easeIn, onComplete:onFadeOut});
}

public
```

```
function set mission( v:MissionVO ):void
{
_mission = v;
_exclamation.visible = !_mission.accepted;
_question.visible = _mission.complete && !_mission.rewardAccepted;
if (_exclamation.visible)
TweenLite.to(_exclamation, .5, {alpha:0.1, ease:Quad.easeOut, onComplete:onFadeOut});
else
TweenLite.killTweensOf(_exclamation);
if (_question.visible)
TweenLite.to(_question, .5, {alpha:0.1, ease:Quad.easeOut, onComplete:onFadeOut});
else
TweenLite.killTweensOf(_question);
}

public function destroy():void
{
if (_exclamation)
TweenLite.killTweensOf(_exclamation);
if (_question)
TweenLite.killTweensOf(_question);
_exclamation = null;
_question = null;
if (onClick)
onClick.removeAll();

onClick = null;

if (_image)
ObjectPool.give(_image);

_image = null;

if (_faqBtn)
_faqBtn.destroy();

_faqBtn = null;

if (_btnText)
_btnText.destroy();

_btnText = null;

_mission = null;
}
}
}
```

File 751: igw\com\ui\hud\shared\bridge\OfferRiverButton.as

```
package
```

```
com.ui.hud.shared.bridge
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.model.player.OfferVO;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.text.TextFormatAlign;
import flash.utils.Timer;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.osflash.signals.Signal;

public class OfferRiverButton extends Sprite
{
public var onClick:Signal;

private var _offerGlow:Bitmap;
private var _offerStarsA:Bitmap;
private var _offerStarsB:Bitmap;

private var _offerTimeRemaining:Label;
private var _btnText:Label;

private var _offerBtn:BitmapButton;

private var _currentOffer:OfferVO;
private var _offers:Vector.<OfferVO>;
private var _timer:Timer;

private var _offerText:String = 'CodeString.OfferWindow.Offer' //OFFER

public function OfferRiverButton()
{
onClick = new Signal(OfferVO);
super();

_offerBtn = UIFactory.getButton(ButtonEnum.OFFER_CHEST);
_offerBtn.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);
_offerBtn.addEventListener(MouseEvent.ROLL_OVER, onRollOver, false, 0, true);
_offerBtn.addEventListener(MouseEvent.ROLL_OUT, onRollOut, false, 0, true)
_offerBtn.enabled = false;

_offerGlow
```

```
= UIFactory.getBitmap('BeginnersChestGlowBMD');
_offerGlow.alpha = 0.0;
_offerGlow.visible = false;

_offerStarsA = UIFactory.getBitmap('BeginnersChestStarsABMD');
_offerStarsA.alpha = 0.0;
_offerStarsA.visible = false;

_offerStarsB = UIFactory.getBitmap('BeginnersChestStarsBBMD');
_offerStarsB.alpha = 1.0;
_offerStarsB.visible = false;

_offerTimeRemaining = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DYNAMIC, 150,
10);
_offerTimeRemaining.constrictTextToSize = false;
_offerTimeRemaining.align = TextFormatAlign.CENTER;

_btnText = UIFactory.getLabel(LabelEnum.H3, 100, 25);
_btnText.constrictTextToSize = false;
_btnText.text = _offerText;

_timer = new Timer(1000);
_timer.addEventListener(TimerEvent.TIMER, onTimerTick, false, 0, true);

addChild(_offerBtn);
addChild(_offerTimeRemaining);
addChild(_offerGlow);
addChild(_offerStarsA);
addChild(_offerStarsB);
addChild(_btnText);

layout();
}

private function layout():void
{
var btnTextOffset:Number = 16;
if (_offerBtn)
{
_offerBtn.x = 18;
_offerBtn.y = -7;
}

if (_offerGlow)
{
_offerGlow.x = _offerBtn.x;
_offerGlow.y = _offerBtn.y;
}

if (_offerStarsA)
{
```

```
_offerStarsA.x = _offerBtn.x;
_offerStarsA.y = _offerBtn.y;
}

if (_offerStarsB)
{
_offerStarsB.x = _offerBtn.x;
_offerStarsB.y = _offerBtn.y;
}

if (_offerTimeRemaining)
{
_offerTimeRemaining.x = _offerBtn.x + (_offerBtn.width - _offerTimeRemaining.width) * 0.5;
_offerTimeRemaining.y = _offerBtn.height - 17;

btnTextOffset = 3;
}

if (_btnText)
{
_btnText.x = _offerBtn.x + (_offerBtn.width - _btnText.width) * 0.5;
_btnText.y = _offerBtn.height - btnTextOffset;
}
}


private function onMouseClick( e:MouseEvent ):void
{
if (onClick)
onClick.dispatch(_currentOffer);
e.stopPropagation();
}

private function onRollOver( e:MouseEvent = null ):void
{
if (_offerGlow)
{
TweenLite.killTweensOf(_offerGlow);
_offerGlow.alpha = 1;
}
}

private function onRollOut( e:MouseEvent = null ):void
{
if (_offerGlow)
{
_offerGlow.alpha = 0;
onFadeOut(_offerGlow, 1.0, 0.8, 0.1);
}
}
```

```
private function onFadeOut( fadeBitmap:Bitmap, fadeTime:Number, alphaIn:Number = 1.0,
alphaOut:Number = 0.0 ):void
{
TweenLite.to(fadeBitmap, fadeTime, {alpha:alphaIn, ease:Quad.easeOut,
onComplete:onFadeIn, onCompleteParams:[fadeBitmap, fadeTime, alphaIn, alphaOut],
overwrite:0});
}

private function onFadeIn( fadeBitmap:Bitmap, fadeTime:Number, alphaIn:Number = 1.0,
alphaOut:Number = 0.0 ):void
{
TweenLite.to(fadeBitmap, fadeTime, {alpha:alphaOut, ease:Quad.easeIn,
onComplete:onFadeOut, onCompleteParams:[fadeBitmap, fadeTime, alphaIn, alphaOut],
overwrite:0});
}

private function onTimerTick( e:TimerEvent ):void
{
if (_currentOffer == null)
{
if (_timer)
_timer.stop();

return;
}
var timeRemaining:Number = _currentOffer.timeRemainingMS;

if (timeRemaining <= 0)
{
_timer.stop();
onOfferEnded();
} else
{
if (_offerTimeRemaining)
_offerTimeRemaining.setBuildTime(timeRemaining / 1000, 3);
}
}

private function onOfferEnded():void
{
if (_offers)
{
var len:uint = _offers.length;
var currentOffer:OfferVO;
for (var i:uint = 0; i < len; ++i)
{
currentOffer = _offers[i];
if (currentOffer.offerPrototype == _currentOffer.offerPrototype)
{
```

```
_offers.splice(i, 1);
_currentOffer = null;
break;
}
}

if (_offerTimeRemaining)
_offerTimeRemaining.text = '';

if (_offers.length > 0)
{
_currentOffer = _offers[0];
if (_timer)
{
_timer.reset();
_timer.start();
}

if (_offerTimeRemaining)
_offerTimeRemaining.visible = true;
} else
{
if (_offerBtn)
_offerBtn.enabled = false;

if (_offerGlow)
{
TweenLite.killTweensOf(_offerGlow);
_offerGlow.visible = false;
}

if (_offerStarsA)
{
TweenLite.killTweensOf(_offerStarsA);
_offerStarsA.visible = false;
}

if (_offerStarsB)
{
TweenLite.killTweensOf(_offerStarsB);
_offerStarsB.visible = false;
}

if (_offerTimeRemaining)
_offerTimeRemaining.visible = false;

}
}
}

public
```

```
function set offers( v:Vector.<OfferVO> ):void
{
_offers = v;
if (_offers && _offers.length > 0)
{
_currentOffer = _offers[0];

if (_offerBtn)
_offerBtn.enabled = true;

if (_offerGlow)
{
TweenLite.killTweensOf(_offerGlow);
_offerGlow.visible = true;
onFadeOut(_offerGlow, 1.0, 0.8, 0.1);
}

if (_offerStarsA)
{
TweenLite.killTweensOf(_offerStarsA);
_offerStarsA.visible = true;
onFadeOut(_offerStarsA, 1.0);
}

if (_offerStarsB)
{
TweenLite.killTweensOf(_offerStarsB);
_offerStarsB.visible = true;
onFadeIn(_offerStarsB, 1.0);
}

if (_btnText)
{
_btnText.bold = true;
_btnText.textColor = 0xd1e5f7;
_btnText.text = _offerText;
_btnText.x = _offerBtn.x + (_offerBtn.width - _btnText.width) * 0.5;
_btnText.y = _offerBtn.height - 3;
}

if (_timer)
{
_timer.reset();
_timer.start();
}

if (_offerTimeRemaining)
_offerTimeRemaining.visible = true;

} else
{
```

```
if (_offerBtn)
_offerBtn.enabled = false;

if (_offerGlow)
{
TweenLite.killTweensOf(_offerGlow);
_offerGlow.visible = false;
}

if (_offerStarsA)
{
TweenLite.killTweensOf(_offerStarsA);
_offerStarsA.visible = false;
}

if (_btnText)
{
_btnText.bold = false;
_btnText.textColor = 0x213745;
_btnText.text = _offerText;
_btnText.x = _offerBtn.x + (_offerBtn.width - _btnText.width) * 0.5;
_btnText.y = _offerBtn.height - 16;
}

if (_offerStarsB)
{
TweenLite.killTweensOf(_offerStarsB);
_offerStarsB.visible = false;
}

if (_offerTimeRemaining)
_offerTimeRemaining.visible = false;
}
}

public function get currentOffer():OfferVO
{
return _currentOffer;
}

public function destroy():void
{
if (onClick)
onClick.removeAll();

onClick = null;

if (_offerBtn)
{
_offerBtn.removeEventListener(MouseEvent.CLICK,
```

```
onMouseClick);
_offerBtn.removeEventListener(MouseEvent.ROLL_OVER, onRollOver);
_offerBtn.removeEventListener(MouseEvent.ROLL_OUT, onRollOut)
_offerBtn.destroy();
}

_offerBtn = null;

if (_offerGlow)
TweenLite.killTweensOf(_offerGlow);

_offerGlow = null;

if (_offerStarsA)
TweenLite.killTweensOf(_offerStarsA);

_offerStarsA = null;

if (_offerStarsB)
TweenLite.killTweensOf(_offerStarsB);

_offerStarsB = null;

if (_timer)
{
_timer.removeEventListener(TimerEvent.TIMER, onTimerTick);
if (_timer.running)
_timer.stop();
}
_timer = null;

if (_offerTimeRemaining)
_offerTimeRemaining.destroy();

_offerTimeRemaining = null;

if (_btnText)
_btnText.destroy();

_btnText = null;

}
}
}

-----------------------------------
File 752: igw\com\ui\hud\shared\command\CommandView.as
package com.ui.hud.shared.command
{
import com.Application;
import
```

```
com.enum.PositionEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.StateEvent;
import com.presenter.shared.ICommandPresenter;
import com.ui.UIFactory;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.tab.TabComponent;
import com.ui.core.effects.EffectFactory;

import flash.events.Event;
import flash.events.MouseEvent;

import org.greensock.TweenLite;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class CommandView extends View
{
public static const TRADE_TAB:String = "TradeRoutesTab";
public static const FLEET_TAB:String = "FleetTab";

private const MAXIMIZED:Number = 1;
private const MIN_X_POS:Number = 271;
private const MIN_Y_POS:Number = 452;
private const MINIMIZED:Number = 0;

private var _minimizeButton:BitmapButton;
private var _maximizeButton:BitmapButton;
private var _selectedView:View;
private var _tabComponent:TabComponent;
private var _windowState:int;

private var _fleetView:FleetCommandView;
private var _tradeView:ResourceTradeCommandView;

private var _resourcesTabText:String = 'CodeString.CommandView.Resources'; //RESOURCES
private var _fleetsTabText:String = 'CodeString.CommandView.Fleets'; //FLEETS


[PostConstruct]
override public function init():void
{
super.init();
_windowState = MAXIMIZED;

var headerSize:int = 32;
_tabComponent
```

```
= ObjectPool.get(TabComponent);
_tabComponent.init(PanelEnum.CONTAINER_DOUBLE_NOTCHED_ARROWS,
PanelEnum.HEADER_NOTCHED, 355, 210, headerSize);
_tabComponent.automaticLayoutHorizontal = true;
_tabComponent.addTab(TRADE_TAB, ButtonEnum.HEADER_NOTCHED, 126, headerSize, 0,
0, _resourcesTabText, LabelEnum.H2);
_tabComponent.addTab(FLEET_TAB, ButtonEnum.HEADER, 100, headerSize, 0, 0,
_fleetsTabText, LabelEnum.H2);
_tabComponent.addSwitchTabListener(onTabSwitched);

_minimizeButton = UIFactory.getButton(ButtonEnum.ICON_WINDOW_MIN, 0, 0, 330, 13);
_maximizeButton = UIFactory.getButton(ButtonEnum.ICON_WINDOW_FULL, 0, 0, 330, 13);
_maximizeButton.visible = false;

_tradeView = new ResourceTradeCommandView();
_tradeView.y = headerSize;

_fleetView = new FleetCommandView();
_fleetView.y = headerSize;

presenter.injectObject(_tradeView);
presenter.injectObject(_fleetView);

addListener(_minimizeButton, MouseEvent.CLICK, onMinimize);
addListener(_maximizeButton, MouseEvent.CLICK, onMaximize);

addChild(_tabComponent);
addChild(_minimizeButton);
addChild(_maximizeButton);
addChild(_tradeView);
addChild(_fleetView);

onStateChange(Application.STATE);

onStageResized();
addHitArea();
addEffects();
effectsIN();

visible = !presenter.inFTE;
}

private function onTabSwitched( name:String ):void
{
if (name == TRADE_TAB)
{
_tradeView.visible = true;
_fleetView.visible = false;
} else
{
_fleetView.visible
```

```
= true;
_tradeView.visible = false;
}

}

override protected function onStateChange( state:String ):void
{
if (state == StateEvent.GAME_SECTOR)
_tabComponent.setSelectedTab(FLEET_TAB);
else if (state == StateEvent.GAME_STARBASE)
_tabComponent.setSelectedTab(TRADE_TAB);
if (state == StateEvent.GAME_BATTLE)
destroy();
}

private function onMinimize( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
_windowState = MINIMIZED;
_minimizeButton.visible = false;
_maximizeButton.visible = true;
TweenLite.to(this, .2, {y:DeviceMetrics.HEIGHT_PIXELS - (32 * Application.SCALE)});
}

private function onMaximize( e:MouseEvent ):void
{
if (!presenter.hudEnabled)
return;
_windowState = MAXIMIZED;
_minimizeButton.visible = true;
_maximizeButton.visible = false;
TweenLite.to(this, .2, {y:DeviceMetrics.HEIGHT_PIXELS - height});
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.LEFT, PositionEnum.BOTTOM,
onStageResized));
}

private function onStageResized( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
TweenLite.killTweensOf(this);
var yPos:Number;
switch (_windowState)
{
case MAXIMIZED:
yPos
```

```
= DeviceMetrics.HEIGHT_PIXELS - height;
break;
case MINIMIZED:
yPos = DeviceMetrics.HEIGHT_PIXELS - (32 * Application.SCALE);
break;
}
y = (yPos < MIN_Y_POS) ? MIN_Y_POS : yPos;
x = (DeviceMetrics.WIDTH_PIXELS - width < MIN_X_POS) ? MIN_X_POS :
DeviceMetrics.WIDTH_PIXELS - width;
if (_fleetView)
_fleetView.layoutFleets();
}

override public function get height():Number { return _tabComponent ? _tabComponent.height *
Application.SCALE : this.height * Application.SCALE; }
override public function get width():Number { return _tabComponent ? _tabComponent.width *
Application.SCALE : this.width * Application.SCALE; }

[Inject]
public function set presenter( value:ICommandPresenter ):void { _presenter = value; }
public function get presenter():ICommandPresenter { return ICommandPresenter(_presenter); }

override public function get type():String { return ViewEnum.UI }

override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
TweenLite.killTweensOf(this);

super.destroy();

_maximizeButton = UIFactory.destroyButton(_maximizeButton);
_minimizeButton = UIFactory.destroyButton(_minimizeButton);

ObjectPool.give(_tabComponent);
_tabComponent = null;

if (_fleetView)
_fleetView.destroy();

_fleetView = null;

if (_tradeView)
_tradeView.destroy();

_tradeView = null;
}
}
}

-----------------------------------
```

File 753: igw\com\ui\hud\shared\command\FleetButton.as

```
package com.ui.hud.shared.command
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.fleet.FleetVO;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class FleetButton extends Sprite
{
private static const NORMAL:int = 0;
private static const OTHER_SECTOR:int = 1;
private static const BATTLE:int = 2;

public var onLoadSmallImage:Signal;

private var _name:Label;
private var _cargo:Label;
private var _healthBar:ProgressBar;
private var _fleetID:String;
private var _transgateIcon:Bitmap;
private var _cargoImage:Bitmap;
private var _shipFrame:ScaleBitmap;
private var _btn:BitmapButton;
private var _image:ImageComponent;
private var _fleetVO:FleetVO;
private var _currentSector:String;
private var _btnSkins:Array;
private var _state:int;

public function FleetButton( skins:Array ):void
{
onLoadSmallImage = new Signal(String, Function);

_btnSkins = skins;

_btn
```

```
= UIFactory.getButton(ButtonEnum.FRAME_BLUE, 186, 60);
_btn.selectable = true;

_image = ObjectPool.get(ImageComponent);
_image.init(48, 47);

_name = UIFactory.getLabel(LabelEnum.DEFAULT, 125, 40, 6, 6);
_name.fontSize = 18;
_name.constrictTextToSize = true;
_name.align = TextFormatAlign.LEFT;
addChild(_name);

_cargo = new Label(14, 0xf0f0f0, 50, 20, false);
_cargo.align = TextFormatAlign.LEFT;
_cargo.x = 99;
_cargo.y = 36;
addChild(_cargo);

_shipFrame = UIFactory.getScaleBitmap(PanelEnum.CHARACTER_FRAME);
_shipFrame.width = 48;
_shipFrame.height = 47;
_shipFrame.x = 132;
_shipFrame.y = 7;

_cargoImage = UIFactory.getBitmap('CargoIconBMD');
_cargoImage.x = 72;
_cargoImage.y = 34;

_healthBar = new ProgressBar();
_healthBar.init(ProgressBar.VERTICAL, new Bitmap(new BitmapData(42, 37, true,
0x8Cff0000)), null, 0.01);
_healthBar.x = _shipFrame.x + 3;
_healthBar.y = _shipFrame.y + 6;
_healthBar.setMinMax(0, 1);

_state = 0;

addChild(_btn);
addChild(_image);
addChild(_healthBar);
addChild(_shipFrame);
addChild(_cargoImage);
addChild(_cargo);
addChild(_name);
}

public function setFleetData( fleetVO:FleetVO, currentSector:String ):void
{
if (fleetVO)
{
_currentSector
```

```
= currentSector;
_fleetVO = fleetVO;
_fleetID = _fleetVO.id;

_healthBar.amount = 1 - fleetVO.currentHealth;
_cargo.text = _fleetVO.cargoPercent + '%';
_name.text = _fleetVO.name;

onLoadSmallImage.dispatch(fleetVO.asset, onImageLoaded);

if (_fleetVO.inBattle)
{
if (_state != BATTLE)
_btn.updateBackgrounds(_btnSkins[8], _btnSkins[9], _btnSkins[10], null, _btnSkins[11]);
_state = BATTLE;
} else
{
if (_currentSector == _fleetVO.sector)
{
if (_state != NORMAL)
_btn.updateBackgrounds(_btnSkins[0], _btnSkins[1], _btnSkins[2], null, _btnSkins[3]);
_state = NORMAL;
} else
{
if (_state != OTHER_SECTOR)
_btn.updateBackgrounds(_btnSkins[4], _btnSkins[5], _btnSkins[6], null, _btnSkins[7]);
_state = OTHER_SECTOR
}
}
} else
_image.clearBitmap();

}

private function onImageLoaded( asset:BitmapData ):void
{
if (_image && _shipFrame)
{
_image.onImageLoaded(asset);
_image.x = _shipFrame.x + (_shipFrame.width - _image.width) * 0.5;
_image.y = _shipFrame.y + (_shipFrame.height - _image.height) * 0.5;
}
}

public function get fleetName():String { return _fleetVO.name; }

public function set selected( v:Boolean ):void { _btn.selected = v; }
public function get selected():Boolean { return _btn.selected; }

public function get fleet():FleetVO { return _fleetVO; }
public
```

```
function get fleetID():String { return _fleetID; }

public function get inBattle():Boolean { return _fleetVO.inBattle }

public function get inSector():Boolean { return (_currentSector == _fleetVO.sector); }

public function destroy():void
{
if (onLoadSmallImage)
onLoadSmallImage.removeAll();

onLoadSmallImage = null;

if (_name)
_name.destroy();

_name = null;

if (_cargo)
_cargo.destroy();

_cargo = null;


if (_healthBar)
_healthBar.destroy();

if (_btn)
_btn.destroy();

if (_image)
ObjectPool.give(_image);

_btnSkins.length = 0;

_image = null;

_healthBar = null;

_transgateIcon = null;
_cargoImage = null;
_shipFrame = null;
_state = 3;
}

}
}
```

File 754: igw\com\ui\hud\shared\command\FleetCommandView.as

```
package
```

```
com.ui.hud.shared.command
{
import com.Application;
import com.enum.ui.PanelEnum;
import com.event.StateEvent;
import com.game.entity.components.shared.Cargo;
import com.game.entity.components.shared.Move;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.presenter.shared.CommandPresenter;
import com.presenter.shared.ICommandPresenter;
import com.ui.UIFactory;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;
import com.ui.modal.dock.ShipIcon;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.Timer;

import org.adobe.utils.StringUtil;
import org.ash.core.Entity;


public class FleetCommandView extends Sprite
{
private var _fleetFrame:Bitmap;
private var _fleetInfoFrame:Bitmap;

private var _fleetBtnStates:Array;

private var _selectedFleet:FleetButton;
private var _activeFleets:Vector.<FleetButton>;
private var _lookup:Dictionary;

private var _selectedFleetShips:Vector.<ShipIcon>;

private var _fleetName:Label;

private var _cargoCountTitle:Label;
private var _targetTitle:Label;
private var _statusTitle:Label;

private var _cargoCount:Label;
private var _target:Label;
private var _status:Label;
private
```

```
var _travelTime:Label;

private var _currentState:String;
private var _travelTimeRemaining:int;
private var _travelTimer:Timer;

private var _presenter:ICommandPresenter;

private const STATUS_TEXT:String = 'CodeString.FleetSelection.State'; //Status
private const TARGET_TEXT:String = 'CodeString.FleetSelection.Target'; //Target
private const CARGO_TEXT:String = 'CodeString.FleetSelection.Cargo'; //Cargo
private const FLEET_STATUS_MOVING:String = 'CodeString.Fleet.Status.Moving'; //Moving
private const FLEET_STATUS_IDLE:String = 'CodeString.Fleet.Status.Idle'; //Idle
private const FLEET_STATUS_BATTLE:String = 'CodeString.Fleet.Status.Battle'; //In Battle
private const SECTOR_RAW_COORDS:String = 'CodeString.Sector.RawCoords';
//[[Number.CoordinateX]] , [[Number.CoordinateY]]
private const CARGO_CAPACITY:String = 'CodeString.Sector.CargoCapacity';
//[[Number.CurrentCargoCount]] / [[Number.MaxCargoCount]] ([[Number.CargoPercent]]%)
private const SHARED_NONE:String = 'CodeString.Shared.None'; //None
private const _fleetDefendingText:String = 'CodeString.Alert.Battle.DefendingStatus';
//Defending

public function FleetCommandView():void
{
_activeFleets = new Vector.<FleetButton>();
_lookup = new Dictionary();
_selectedFleetShips = new Vector.<ShipIcon>();

_fleetBtnStates = new Array();
_fleetBtnStates.push(UIFactory.getBitmapData('BtnEmptyUpBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnEmptyROBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnEmptyDownBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnEmptySelectedBMD'))

_fleetBtnStates.push(UIFactory.getBitmapData('BtnGoldUpBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnGoldROBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnGoldDownBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnGoldSelectedBMD'))

_fleetBtnStates.push(UIFactory.getBitmapData('BtnRepairUpBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnRepairROBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnRepairDownBMD'))
_fleetBtnStates.push(UIFactory.getBitmapData('BtnRepairSelectedBMD'))

_fleetFrame = UIFactory.getBitmap('SectorFleetSelectionBGBMD');

_fleetInfoFrame = UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL,
191, 163);

_fleetName = new Label(16, 0xb4e0ff, 250, 30);
_fleetName.useLocalization
```

```
= false;
_fleetName.textColor = 0xb4e0ff;
_fleetName.align = TextFormatAlign.LEFT;

_statusTitle = new Label(16, 0xb4e0ff, 111);
_statusTitle.constrictTextToSize = false;
_statusTitle.align = TextFormatAlign.LEFT;
_statusTitle.text = STATUS_TEXT;

_status = new Label(16, 0xf0f0f0);
_status.bold = true;
_status.constrictTextToSize = false;
_status.align = TextFormatAlign.LEFT;

_targetTitle = new Label(16, 0xb4e0ff, 111);
_targetTitle.constrictTextToSize = false;
_targetTitle.align = TextFormatAlign.LEFT;
_targetTitle.text = TARGET_TEXT;

_target = new Label(16, 0xf0f0f0);
_target.bold = true;
_target.constrictTextToSize = false;
_target.align = TextFormatAlign.LEFT;

_cargoCountTitle = new Label(16, 0xb4e0ff, 111);
_cargoCountTitle.constrictTextToSize = false;
_cargoCountTitle.align = TextFormatAlign.LEFT;
_cargoCountTitle.text = CARGO_TEXT;

_cargoCount = new Label(16, 0xf0f0f0, 220);
_cargoCount.bold = true;
_cargoCount.constrictTextToSize = false;
_cargoCount.align = TextFormatAlign.LEFT;
_cargoCount.useLocalization = false;

_travelTime = new Label(16, LabelFactory.DYNAMIC_TEXT_COLOR, 60, 20, false);
_travelTime.constrictTextToSize = false;
_travelTime.align = TextFormatAlign.CENTER;
_travelTime.useLocalization = false;

addChild(_fleetFrame);
addChild(_fleetInfoFrame);
addChild(_fleetName);
addChild(_statusTitle);
addChild(_status);
addChild(_targetTitle);
addChild(_target);
addChild(_cargoCountTitle);
addChild(_cargoCount);
addChild(_travelTime);

var
```

```
currentShip:ShipIcon;
var xPos:Number;
var yPos:Number;
for (var i:uint = 0; i < 6; ++i)
{
currentShip = new ShipIcon();
currentShip.scale(0.37, 0.37);
currentShip.mouseEnabled = false;
addChild(currentShip);
_selectedFleetShips.push(currentShip);
}

_travelTimer = new Timer(1000);
_travelTimer.addEventListener(TimerEvent.TIMER, onTravelTimerTick, false, 0, true);
}

[PostConstruct]
public function init():void
{
_presenter && _presenter.highfive();
presenter.addListenerForFleetUpdate(showFleets);
presenter.addStateListener(onStateChange);

if (Application.STATE == StateEvent.GAME_SECTOR)
presenter.addSelectionChangeListener(onSelectionChange);

showFleets();
layout();
}

public function layout():void
{
_fleetInfoFrame.x = 155;
_fleetInfoFrame.y = 39;

_fleetFrame.x = 10;
_fleetFrame.y = 31;

_fleetName.x = 150;
_fleetName.y = 12;

_statusTitle.x = 159;
_statusTitle.y = 44;

_status.x = 159;
_status.y = 65;

_targetTitle.x = 159;
_targetTitle.y = 96;

_target.x
```

```
= 159;
_target.y = 121;

_cargoCountTitle.x = 159;
_cargoCountTitle.y = 149;

_cargoCount.x = 159;
_cargoCount.y = 171;

_travelTime.x = 50;
_travelTime.y = 93;

var len:uint = _selectedFleetShips.length;
var currentShipIcon:ShipIcon;
var xPos:Number;
var yPos:Number;
for (var i:uint = 0; i < len; ++i)
{
currentShipIcon = _selectedFleetShips[i];
switch (i)
{
case 0:
xPos = 56;
yPos = 31;
break;
case 1:
xPos = 10;
yPos = 56;
break;
case 2:
xPos = 100;
yPos = 56;
break;
case 3:
xPos = 10;
yPos = 108;
break;
case 4:
xPos = 100;
yPos = 108;
break;
case 5:
xPos = 55;
yPos = 135;
break;
}
currentShipIcon.x = xPos;
currentShipIcon.y = yPos;
}
}

public
```

```
function showFleets( vo:FleetVO = null ):void
{
var fleets:Vector.<FleetVO> = presenter.fleets;
var focusFleetID:String = presenter.focusFleetID;
var button:FleetButton;
var currentFleet:FleetVO;
var sector:String = presenter.sectorID;
for (var i:int = 0; i < fleets.length; i++)
{
button = null;
if (!(fleets[i].id in _lookup) && fleets[i].sector != '')
button = createFleetButton(fleets[i]);
else if (_lookup[fleets[i].id] && fleets[i].sector == '')
removeFleetButton(fleets[i]);
else
button = _lookup[fleets[i].id];

if (button)
{
currentFleet = fleets[i];
button.setFleetData(currentFleet, sector);
if (!(fleets[i].id in _lookup))
_lookup[button.fleetID] = button;
}

if (_selectedFleet == null && (focusFleetID == null || fleets[i].id == focusFleetID) &&
fleets[i].sector == presenter.sectorID)
onClick(button, (!_selectedFleet && fleets[i].id == focusFleetID), false);
}

if (vo != null && _selectedFleet && vo.id == _selectedFleet.fleetID)
{
if (_travelTimer.running)
_travelTimer.stop();
}

updateSelectedFleetStats();
layoutFleets();
}

private function createFleetButton( fleetVO:FleetVO ):FleetButton
{
//add a new fleet button
var button:FleetButton = new FleetButton(_fleetBtnStates);
button.addEventListener(MouseEvent.CLICK, onFleetClicked, false, 0, true);
button.onLoadSmallImage.add(presenter.loadSmallImageFromEntityData);
addChild(button);
_activeFleets.push(button);

return button;
}
```

```
private function removeFleetButton( fleetVO:FleetVO ):void
{
//remove a fleet button
for (var j:int = 0; j < _activeFleets.length; j++)
{
if (_activeFleets[j].fleetID == fleetVO.id)
{
_activeFleets[j].removeEventListener(MouseEvent.CLICK, onFleetClicked);
removeChild(_activeFleets[j]);
_activeFleets.splice(j, 1);
_lookup[fleetVO.id] = null;
delete _lookup[fleetVO.id];
if (_selectedFleet && _selectedFleet.fleetID == fleetVO.id)
{
if (_activeFleets.length == 0)
_selectedFleet = null;
else
_selectedFleet = _activeFleets[0];
}
break;
}
}
}

private function updateSelectedFleetStats():void
{
var len:uint;
var currentImage:ShipIcon;
var i:uint

if (_selectedFleet)
{
var fleetVO:FleetVO = presenter.getFleetVO(_selectedFleet.fleetID);
var ships:Vector.<ShipVO> = fleetVO.ships;
len = ships.length;
var currentShip:ShipVO;
for (i = 0; i < len; ++i)
{
currentShip = ships[i];
currentImage = _selectedFleetShips[i];
currentImage.clearImageBitmap();
if (currentShip)
{
currentImage.onLoadShipImage.add(presenter.loadIconImageFromEntityData);
currentImage.setShip(currentShip, null, fleetVO);
} else
currentImage.setShip(null);
}
_fleetName.text
```

```
= _selectedFleet.fleetName;
_selectedFleet.setFleetData(fleetVO, presenter.sectorID);
var entity:Entity = presenter.getEntity(_selectedFleet.fleetID);
var move:Move = entity ? entity.get(Move) : null;
var cargo:Cargo = entity ? entity.get(Cargo) : null;
var currentCargo:int = cargo ? cargo.cargo : fleetVO.currentCargo;
var cargoPercent:Number = currentCargo > 0 ? int(currentCargo / fleetVO.maxCargo * 100) : 0;

_cargoCount.setTextWithTokens(CARGO_CAPACITY,
{'[[Number.CurrentCargoCount]]':StringUtil.commaFormatNumber(currentCargo),
'[[Number.MaxCargoCount]]':StringUtil.commaFormatNumber(fleetVO.maxCargo),
'[[Number.CargoPercent]]':cargoPercent});

var rawCoordsDict:Dictionary;
_travelTime.visible = false;
if (fleetVO.inBattle)
{
_status.text = FLEET_STATUS_BATTLE;
_target.setTextWithTokens(SECTOR_RAW_COORDS,
{'[[Number.CoordinateX]]':int(fleetVO.sectorLocationX * 0.01),
'[[Number.CoordinateY]]':int(fleetVO.sectorLocationY * 0.01)});
} else if (move && move.moving)
{
_status.text = FLEET_STATUS_MOVING;
_target.setTextWithTokens(SECTOR_RAW_COORDS,
{'[[Number.CoordinateX]]':int(move.destination.x * 0.01),
'[[Number.CoordinateY]]':int(move.destination.y * 0.01)});

_travelTimeRemaining = move.totalTime - move.time;
_travelTime.setBuildTime(_travelTimeRemaining, 2);
_travelTime.visible = true;
_travelTimer.start();


} else
{
_status.text = fleetVO.defendTarget != "" ? _fleetDefendingText : FLEET_STATUS_IDLE;
_target.text = fleetVO.defendTarget != "" ? "Starbase" : SHARED_NONE; //change this later to
the type of object we're defending when we can defend more than a starbase
}

} else
{
_fleetName.text = '';
_cargoCount.text = '';
_status.text = '';
_target.text = '';
len = _selectedFleetShips.length;
for (i = 0; i < len; ++i)
{
currentImage
```

```
= _selectedFleetShips[i];
currentImage.setShip(null);
}
}
}

private function onTravelTimerTick( e:TimerEvent ):void
{
_travelTimeRemaining -= 1;
if (_travelTimeRemaining > 0)
_travelTime.setBuildTime(_travelTimeRemaining, 2);
else
_travelTimer.stop();
}

private function onFleetClicked( e:MouseEvent ):void
{
if (e)
{
e.stopImmediatePropagation();
onClick(FleetButton(e.currentTarget));
}
}

private function onClick( clickedFleet:FleetButton, gotoLocation:Boolean = true,
canEnterBattle:Boolean = true, informPresenter:Boolean = true ):void
{
if (!presenter.hudEnabled)
return;
if (clickedFleet != null)
{
if (_selectedFleet != null)
_selectedFleet.selected = false;

gotoLocation = ((_selectedFleet != null && (_selectedFleet.fleetID == clickedFleet.fleetID ||
!clickedFleet.inSector) && gotoLocation)) ? gotoLocation : false;

canEnterBattle = ((_selectedFleet != null && _selectedFleet.fleetID == clickedFleet.fleetID &&
canEnterBattle)) ? canEnterBattle : false;

_selectedFleet = clickedFleet;
_selectedFleet.selected = true;

updateSelectedFleetStats();
if (informPresenter)
presenter.selectFleet(_selectedFleet.fleetID, gotoLocation, canEnterBattle);
}
}

private function onSelectionChange( entity:Entity ):void
{
```

```
if (entity)
{
for (var i:int = 0; i < _activeFleets.length; i++)
{
if (_activeFleets[i].fleetID == entity.id)
onClick(_activeFleets[i], false, false, false);
}
}
}

protected function onStateChange( v:String ):void
{
_currentState = v;

switch (_currentState)
{
case StateEvent.GAME_SECTOR:
presenter.addSelectionChangeListener(onSelectionChange);
if (presenter.focusFleetID == null && _selectedFleet != null && _selectedFleet.inSector)
presenter.selectFleet(_selectedFleet.fleetID, false, false);
else if (_selectedFleet != null && _selectedFleet.fleetID != presenter.focusFleetID &&
presenter.focusFleetID in _lookup)
onClick(_lookup[presenter.focusFleetID], false, false, true);
break;
case StateEvent.GAME_SECTOR_CLEANUP:
presenter.removeSelectionChangeListener(onSelectionChange);
break;
case StateEvent.GAME_STARBASE:
showFleets();
if (presenter.focusFleetID == null && _selectedFleet != null)
presenter.selectFleet(_selectedFleet.fleetID, false, false);
break;
}
}

public function layoutFleets():void
{
var yPos:Number = -33;
var xPos:Number = 160;
var secondColumn:Boolean;
var len:uint = _activeFleets.length;
var currentFleet:FleetButton;

for (var i:uint = 0; i < len; ++i)
{
currentFleet = _activeFleets[i];
var vo:FleetVO = currentFleet.fleet;

if (i != 0 && i % 4 == 0)
{
```

```
yPos = -97;
xPos -= 193;
} else
{
yPos -= (currentFleet.height + 3);
}

currentFleet.x = xPos;
currentFleet.y = yPos;
}
mouseEnabled = (len > 0) ? true : false;
}

[Inject]
public function set presenter( value:ICommandPresenter ):void { _presenter = value; }
public function get presenter():ICommandPresenter { return CommandPresenter(_presenter); }

public function destroy():void
{
presenter.removeListenerForFleetUpdate(showFleets);
presenter.removeStateListener(onStateChange);
presenter.removeSelectionChangeListener(onSelectionChange);
_presenter && _presenter.shun();

if (_travelTimer)
{
if (_travelTimer.running)
_travelTimer.stop();

_travelTimer.removeEventListener(TimerEvent.TIMER, onTravelTimerTick);
}
_travelTimer = null;

_fleetFrame = null;
_fleetInfoFrame = null;

_fleetBtnStates.length = 0;

var len:uint = _activeFleets.length;
var i:uint = 0;
var currentFleet:FleetButton;
for (i; i < len; ++i)
{
currentFleet = _activeFleets[i];
currentFleet.removeEventListener(MouseEvent.CLICK, onFleetClicked);
currentFleet.destroy();
currentFleet = null;
}
_activeFleets.length = 0;

_lookup
```

```
= null;

len = _selectedFleetShips.length;
i = 0;
var currentShip:ShipIcon;
for (i; i < len; ++i)
{
currentShip = _selectedFleetShips[i];
currentShip.destroy();
currentShip = null;
}
_selectedFleetShips.length = 0;

if (_fleetName)
_fleetName.destroy();

_fleetName = null;

if (_cargoCountTitle)
_cargoCountTitle.destroy();

_cargoCountTitle = null;

if (_targetTitle)
_targetTitle.destroy();

_targetTitle = null;

if (_statusTitle)
_statusTitle.destroy();

_statusTitle = null;

if (_cargoCount)
_cargoCount.destroy();

_cargoCount = null;

if (_target)
_target.destroy();

_target = null;

if (_status)
_status.destroy();

_status = null;

if (_travelTime)
_travelTime.destroy();

_travelTime
```

```
= null;
}
}
}

-----------------------------------
File 755: igw\com\ui\hud\shared\command\ResourceComponent.as
package com.ui.hud.shared.command
{
import com.enum.CurrencyEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.component.IComponent;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;

import org.adobe.utils.StringUtil;
import org.osflash.signals.Signal;

public class ResourceComponent extends Sprite implements IComponent
{
private var _enabled:Boolean = false;
private var _moreSignal:Signal;

private var _alloyCheck:Bitmap;
private var _creditsCheck:Bitmap;
private var _energyCheck:Bitmap;
private var _syntheticsCheck:Bitmap;

private var _alloyIcon:Bitmap;
private var _creditsIcon:Bitmap;
private var _energyIcon:Bitmap;
private var _syntheticsIcon:Bitmap;

private var _alloyHolder:Sprite;
private var _creditsHolder:Sprite;
private var _energyHolder:Sprite;
private var _syntheticsHolder:Sprite;

private var _alloyBar:ProgressBar;
private var _creditsBar:ProgressBar;
private var _energyBar:ProgressBar;
private var _syntheticsBar:ProgressBar;

private
```

```
var _alloyLabel:Label;
private var _creditsLabel:Label;
private var _energyLabel:Label;
private var _syntheticsLabel:Label;

private var _alloyMore:BitmapButton;
private var _creditsMore:BitmapButton;
private var _energyMore:BitmapButton;
private var _syntheticsMore:BitmapButton;

public function init( showPluses:Boolean, showInnerBar:Boolean, verticalSpacing:int = 48 ):void
{
_moreSignal = new Signal(String);

_alloyHolder = new Sprite();
_alloyHolder.y = verticalSpacing;
_creditsHolder = new Sprite();
_creditsHolder.x = 5;
_energyHolder = new Sprite();
_energyHolder.x = 162;
_syntheticsHolder = new Sprite();
_syntheticsHolder.x = 158;
_syntheticsHolder.y = verticalSpacing;

_alloyIcon = UIFactory.getPanel("IconAlloyBMD", 0, 0);
_alloyIcon.smoothing = true;
_creditsIcon = UIFactory.getPanel("IconCreditBMD", 0, 0);
_creditsIcon.smoothing = true;
_energyIcon = UIFactory.getPanel("IconEnergyBMD", 0, 0);
_energyIcon.smoothing = true;
_syntheticsIcon = UIFactory.getPanel("IconSynthBMD", 0, 0);
_syntheticsIcon.smoothing = true;

_alloyBar = UIFactory.getProgressBar(showInnerBar ?
UIFactory.getPanel(PanelEnum.STATBAR, 94, 18) : null,
UIFactory.getBitmap("ResourceBoxBMD"), 0, 1, 1,
_alloyIcon.x + _alloyIcon.width + 2, _alloyIcon.y + 3);
_creditsBar = UIFactory.getProgressBar(showInnerBar ?
UIFactory.getPanel(PanelEnum.STATBAR, 94, 18) : null,
UIFactory.getBitmap("ResourceBoxBMD"), 0, 1, 1,
_creditsIcon.x + _creditsIcon.width + 2, _creditsIcon.y + 3);
_energyBar = UIFactory.getProgressBar(showInnerBar ?
UIFactory.getPanel(PanelEnum.STATBAR, 94, 18) : null,
UIFactory.getBitmap("ResourceBoxBMD"), 0, 1, 1,
_energyIcon.x + _energyIcon.width + 2, _energyIcon.y + 3);
_syntheticsBar = UIFactory.getProgressBar(showInnerBar ?
UIFactory.getPanel(PanelEnum.STATBAR, 94, 18) : null,
UIFactory.getBitmap("ResourceBoxBMD"), 0, 1, 1,
_syntheticsIcon.x + _syntheticsIcon.width + 2, _syntheticsIcon.y + 3);

_alloyLabel
```

```
= UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, 101, 24, _alloyBar.x, _alloyBar.y);
_creditsLabel = UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, 101, 24,
_creditsBar.x, _creditsBar.y);
_energyLabel = UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, 101, 24,
_energyBar.x, _energyBar.y);
_syntheticsLabel = UIFactory.getLabel(LabelEnum.DEFAULT_OPEN_SANS, 101, 24,
_syntheticsBar.x, _syntheticsBar.y);

_alloyHolder.addChild(_alloyIcon);
_creditsHolder.addChild(_creditsIcon);
_energyHolder.addChild(_energyIcon);
_syntheticsHolder.addChild(_syntheticsIcon);

_alloyHolder.addChild(_alloyBar);
_creditsHolder.addChild(_creditsBar);
_energyHolder.addChild(_energyBar);
_syntheticsHolder.addChild(_syntheticsBar);

_alloyHolder.addChild(_alloyLabel);
_creditsHolder.addChild(_creditsLabel);
_energyHolder.addChild(_energyLabel);
_syntheticsHolder.addChild(_syntheticsLabel);

if (showPluses)
{
_alloyMore = UIFactory.getButton(ButtonEnum.PLUS, 0, 0, _alloyBar.x + _alloyBar.width + 4,
_alloyBar.y + 2);
_creditsMore = UIFactory.getButton(ButtonEnum.PLUS, 0, 0, _creditsBar.x + _creditsBar.width
+ 4, _creditsBar.y + 2);
_energyMore = UIFactory.getButton(ButtonEnum.PLUS, 0, 0, _energyBar.x + _energyBar.width
+ 4, _energyBar.y + 2);
_syntheticsMore = UIFactory.getButton(ButtonEnum.PLUS, 0, 0, _syntheticsBar.x +
_syntheticsBar.width + 4, _syntheticsBar.y + 2);

_alloyMore.addEventListener(MouseEvent.CLICK, onMoreClick, false, 0, true);
_creditsMore.addEventListener(MouseEvent.CLICK, onMoreClick, false, 0, true);
_energyMore.addEventListener(MouseEvent.CLICK, onMoreClick, false, 0, true);
_syntheticsMore.addEventListener(MouseEvent.CLICK, onMoreClick, false, 0, true);

_alloyHolder.addChild(_alloyMore);
_creditsHolder.addChild(_creditsMore);
_energyHolder.addChild(_energyMore);
_syntheticsHolder.addChild(_syntheticsMore);

_alloyMore.hitArea = _alloyHolder;
_creditsMore.hitArea = _creditsHolder;
_energyMore.hitArea = _energyHolder;
_syntheticsMore.hitArea = _syntheticsHolder;

_alloyCheck = UIFactory.getPanel('CheckMarkBMD', 0, 0, _alloyMore.x - 3, _alloyMore.height -
12);
```

```
_alloyCheck.visible = false;
_creditsCheck = UIFactory.getPanel('CheckMarkBMD', 0, 0, _creditsMore.x - 3,
_creditsMore.height - 12);
_creditsCheck.visible = false;
_energyCheck = UIFactory.getPanel('CheckMarkBMD', 0, 0, _energyMore.x - 3,
_energyMore.height - 12);
_energyCheck.visible = false;
_syntheticsCheck = UIFactory.getPanel('CheckMarkBMD', 0, 0, _syntheticsMore.x - 3,
_syntheticsMore.height - 12);
_syntheticsCheck.visible = false;

_alloyHolder.addChild(_alloyCheck);
_creditsHolder.addChild(_creditsCheck);
_energyHolder.addChild(_energyCheck);
_syntheticsHolder.addChild(_syntheticsCheck);
} else
{
_energyHolder.x -= 15;
_syntheticsHolder.x -= 15;
}

addChild(_alloyHolder);
addChild(_creditsHolder);
addChild(_energyHolder);
addChild(_syntheticsHolder);
}

public function updateResource( amount:int, max:int, type:String ):void
{
switch (type)
{
case CurrencyEnum.ALLOY:
_alloyLabel.text = StringUtil.commaFormatNumber(amount);
_alloyBar.amount = amount / max;
break;
case CurrencyEnum.CREDIT:
_creditsLabel.text = StringUtil.commaFormatNumber(amount);
_creditsBar.amount = amount / max;
break;
case CurrencyEnum.ENERGY:
_energyLabel.text = StringUtil.commaFormatNumber(amount);
_energyBar.amount = amount / max;
break;
case CurrencyEnum.SYNTHETIC:
_syntheticsLabel.text = StringUtil.commaFormatNumber(amount);
_syntheticsBar.amount = amount / max;
break;
}
}

public
```

```
function updateCost( amount:int, canAfford:Boolean, type:String ):void
{
switch (type)
{
case CurrencyEnum.ALLOY:
_alloyLabel.text = StringUtil.commaFormatNumber(amount);
_alloyLabel.textColor = (canAfford) ? 0x48b53c : 0xea1118;
if (_alloyMore)
{
_alloyCheck.visible = (canAfford) ? true : false;
_alloyMore.visible = (canAfford) ? false : true;
}
break;
case CurrencyEnum.CREDIT:
_creditsLabel.text = StringUtil.commaFormatNumber(amount);
_creditsLabel.textColor = (canAfford) ? 0x48b53c : 0xea1118;
if (_creditsMore)
{
_creditsCheck.visible = (canAfford) ? true : false;
_creditsMore.visible = (canAfford) ? false : true;
}
break;
case CurrencyEnum.ENERGY:
_energyLabel.text = StringUtil.commaFormatNumber(amount);
_energyLabel.textColor = (canAfford) ? 0x48b53c : 0xea1118;
if (_energyMore)
{
_energyCheck.visible = (canAfford) ? true : false;
_energyMore.visible = (canAfford) ? false : true;
}
break;
case CurrencyEnum.SYNTHETIC:
_syntheticsLabel.text = StringUtil.commaFormatNumber(amount);
_syntheticsLabel.textColor = (canAfford) ? 0x48b53c : 0xea1118;
if (_syntheticsMore)
{
_syntheticsCheck.visible = (canAfford) ? true : false;
_syntheticsMore.visible = (canAfford) ? false : true;
}
break;
}
}

public function addMoreListener( listener:Function ):void { _moreSignal.add(listener); }
public function removeMoreListener( listener:Function ):void { _moreSignal.remove(listener); }

private function onMoreClick( e:MouseEvent ):void
{
if (!BitmapButton(e.currentTarget).visible)
return;
switch
```

```
(e.currentTarget)
{
case _alloyMore:
_moreSignal.dispatch(CurrencyEnum.ALLOY);
break;
case _creditsMore:
_moreSignal.dispatch(CurrencyEnum.CREDIT);
break;
case _energyMore:
_moreSignal.dispatch(CurrencyEnum.ENERGY);
break;
case _syntheticsMore:
_moreSignal.dispatch(CurrencyEnum.SYNTHETIC);
break;
}
}

public function get alloyHolder():Sprite { return _alloyHolder; }
public function get creditsHolder():Sprite { return _creditsHolder; }
public function get energyHolder():Sprite { return _energyHolder; }
public function get syntheticHolder():Sprite { return _syntheticsHolder; }

public function get enabled():Boolean { return false; }
public function set enabled( value:Boolean ):void { _enabled = value; }

public function set hideBars( v:Boolean ):void
{
_alloyBar.visible = v;
_creditsBar.visible = v;
_energyBar.visible = v;
_syntheticsBar.visible = v;
}

public function destroy():void
{
while (numChildren > 0)
removeChildAt(0);
while (_alloyHolder.numChildren > 0)
_alloyHolder.removeChildAt(0);
_alloyHolder = null;
while (_creditsHolder.numChildren > 0)
_creditsHolder.removeChildAt(0);
_creditsHolder = null;
while (_energyHolder.numChildren > 0)
_energyHolder.removeChildAt(0);
_energyHolder = null;
while (_syntheticsHolder.numChildren > 0)
_syntheticsHolder.removeChildAt(0);
_syntheticsHolder = null;

_moreSignal.removeAll();
```

```
_moreSignal = null;

_alloyIcon = UIFactory.destroyPanel(_alloyIcon);
_creditsIcon = UIFactory.destroyPanel(_creditsIcon);
_energyIcon = UIFactory.destroyPanel(_energyIcon);
_syntheticsIcon = UIFactory.destroyPanel(_syntheticsIcon);

_alloyBar = UIFactory.destroyProgressBar(_alloyBar);
_creditsBar = UIFactory.destroyProgressBar(_creditsBar);
_energyBar = UIFactory.destroyProgressBar(_energyBar);
_syntheticsBar = UIFactory.destroyProgressBar(_syntheticsBar);

_alloyLabel = UIFactory.destroyLabel(_alloyLabel);
_creditsLabel = UIFactory.destroyLabel(_creditsLabel);
_energyLabel = UIFactory.destroyLabel(_energyLabel);
_syntheticsLabel = UIFactory.destroyLabel(_syntheticsLabel);

if (_alloyMore)
{
_alloyMore.removeEventListener(MouseEvent.CLICK, onMoreClick);
_alloyMore = UIFactory.destroyButton(_alloyMore);
_creditsMore.removeEventListener(MouseEvent.CLICK, onMoreClick);
_creditsMore = UIFactory.destroyButton(_creditsMore);
_energyMore.removeEventListener(MouseEvent.CLICK, onMoreClick);
_energyMore = UIFactory.destroyButton(_energyMore);
_syntheticsMore.removeEventListener(MouseEvent.CLICK, onMoreClick);
_syntheticsMore = UIFactory.destroyButton(_syntheticsMore);
}

_alloyCheck = UIFactory.destroyPanel(_alloyCheck);
_creditsCheck = UIFactory.destroyPanel(_creditsCheck);
_energyCheck = UIFactory.destroyPanel(_energyCheck);
_syntheticsCheck = UIFactory.destroyPanel(_syntheticsCheck);

visible = true;
}
}
}

-----------------------------------
File 756: igw\com\ui\hud\shared\command\ResourceTradeCommandView.as
package com.ui.hud.shared.command
{
import com.enum.CurrencyEnum;
import com.enum.ui.PanelEnum;
import com.model.starbase.BaseVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.TradeRouteVO;
import com.model.transaction.TransactionVO;
import
```

```
com.presenter.shared.CommandPresenter;
import com.presenter.shared.ICommandPresenter;
import com.presenter.starbase.ITradePresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.construction.ConstructionInfoView;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.store.StoreView;
import com.ui.modal.traderoute.overview.TradeRouteOverviewView;

import flash.geom.Point;
import flash.utils.Dictionary;

import org.adobe.utils.StringUtil;
import org.parade.core.IView;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class ResourceTradeCommandView extends View
{
[Inject]
public var tooltip:Tooltips;

[Inject]
public var tradePresenter:ITradePresenter;

//art has declared a universal padding of 4
static private const PADDING:Number = 4;
static private const PBAR_SIZE:Point = new Point(90, 20);

private var _contracts:Array = [];
private var _currentBase:BaseVO;

private var _tradeSectionBkgd:ScaleBitmap;
private var _resrcSectionBkgd:ScaleBitmap;

private var _resTooltips:Vector.<String> =
Vector.<String>(["CodeString.Resources.CreditsTooltip", "CodeString.Resources.AlloyTooltip",
"CodeString.Resources.EnergyTooltip", "CodeString.Resources.SyntheticsTooltip"]);

private var _tradeComp0:TradeRouteComponent;
private var _tradeComp1:TradeRouteComponent;
private var _tradeComp2:TradeRouteComponent;
private var _resourceComponent:ResourceComponent;

[PostConstruct]
override public function init():void
{
```

```
super.init();
presenter.removeStateListener(onStateChange);
tradePresenter.addTransactionListener(onUpdateTradeRouteTransactions);

_currentBase = presenter.currentBase;

if (_currentBase)
_currentBase.onResourcesChange.add(onMoneyTick);

createChildren();
onMoneyTick();

onUpdateTradeRouteTransactions(null);
}

private function onUpdateTradeRouteTransactions( transaction:TransactionVO ):void
{
_contracts = [];

var _routes:Vector.<TradeRouteVO> = tradePresenter.tradeRoutes.concat();
_routes.sort(function( a:TradeRouteVO, b:TradeRouteVO ):int
{
if (!a)
return -1;
if (!b)
return 1;

return a.end > b.end ? -1 : 1;
});

for (var vo:TradeRouteVO, i:int; i < _routes.length; i++)
{
vo = _routes[i];

if (_contracts.length < tradePresenter.maxUnlockedContracts)
{
if (vo.end > 0)
_contracts.unshift(vo);
else
_contracts.push(true);
} else if (_contracts.length < 3)
_contracts.push(null);
}
for (i = 0; i < _contracts.length; i++)
{
if (i>2)
continue;

if (this["_tradeComp" + i] != null)
this["_tradeComp"
```

```
+ i].update(_contracts[i]);
}
}

protected function createChildren():void
{
_tradeSectionBkgd =
UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL, 343, 114, 6, 5);
addChild(_tradeSectionBkgd);

_resrcSectionBkgd = UIFactory.getPanel(PanelEnum.CONTAINER_DOUBLE_NOTCHED, 343,
81, 6, _tradeSectionBkgd.y + _tradeSectionBkgd.height + 4);
addChild(_resrcSectionBkgd);

//////////////////////////
// TRADEROUTE SECTION
//////////////////////////

_tradeComp0 = new TradeRouteComponent();
_tradeComp0.tradePresenter = tradePresenter;
_tradeComp0.init();
_tradeComp0.x = 47;
_tradeComp0.y = 12;
_tradeComp0.onClick.add(onContractClick);
addChild(_tradeComp0);

_tradeComp1 = new TradeRouteComponent();
_tradeComp1.tradePresenter = tradePresenter;
_tradeComp1.init();
_tradeComp1.x = _tradeComp0.x + 100;
_tradeComp1.y = 12;
_tradeComp1.onClick.add(onContractClick);
addChild(_tradeComp1);

_tradeComp2 = new TradeRouteComponent();
_tradeComp2.tradePresenter = tradePresenter;
_tradeComp2.init();
_tradeComp2.x = _tradeComp1.x + 100;
_tradeComp2.y = 12;
_tradeComp2.onClick.add(onContractClick);
addChild(_tradeComp2);

//////////////////////////
// RESOURCE SECTION
//////////////////////////

_resourceComponent = ObjectPool.get(ResourceComponent);
_resourceComponent.init(true, true, 40);
_resourceComponent.x = 20;
_resourceComponent.y = _resrcSectionBkgd.y + 5;
_resourceComponent.addMoreListener(onMoreClicked);
```

```
tooltip.addTooltip(_resourceComponent.alloyHolder, this, getAlloyTooltip, '', 250, 180, 18, true);
tooltip.addTooltip(_resourceComponent.creditsHolder, this, getCreditTooltip, '', 250, 180, 18,
true);
tooltip.addTooltip(_resourceComponent.energyHolder, this, getEnergyTooltip, '', 250, 180, 18,
true);
tooltip.addTooltip(_resourceComponent.syntheticHolder, this, getSyntheticTooltip, '', 250, 180,
18, true);
addChild(_resourceComponent);

onMoneyTick();
}

private function onContractClick( state:String ):void
{
if (!presenter.hudEnabled)
return;
if (state != TradeRouteComponent.STATE_LOCKED)
{
var view:IView = _viewFactory.createView(TradeRouteOverviewView);
_viewFactory.notify(view);
} else
{
var building:BuildingVO = presenter.getBuildingVOByClass('Surveillance');
if (building)
{
var upgradeView:ConstructionInfoView =
ConstructionInfoView(_viewFactory.createView(ConstructionInfoView));
upgradeView.setup(ConstructionView.BUILD, building);
_viewFactory.notify(upgradeView);
}
}
}

private function onMoneyTick():void
{
_resourceComponent.updateResource(presenter.currentBase.alloy,
presenter.currentBase.maxResources, CurrencyEnum.ALLOY);
_resourceComponent.updateResource(presenter.currentBase.credits,
presenter.currentBase.maxCredits, CurrencyEnum.CREDIT);
_resourceComponent.updateResource(presenter.currentBase.energy,
presenter.currentBase.maxResources, CurrencyEnum.ENERGY);
_resourceComponent.updateResource(presenter.currentBase.synthetic,
presenter.currentBase.maxResources, CurrencyEnum.SYNTHETIC);
}

private function getCreditTooltip():String
{
var resourceTooltipDict:Dictionary = new Dictionary();
resourceTooltipDict['[[Number.BaseResourceCount]]']
```

```
= StringUtil.commaFormatNumber(_currentBase.credits);
resourceTooltipDict['[[Number.MaxResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.maxCredits);

return Localization.instance.getStringWithTokens(_resTooltips[0], resourceTooltipDict);
}

private function getAlloyTooltip():String
{
var resourceTooltipDict:Dictionary = new Dictionary();
resourceTooltipDict['[[Number.BaseResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.alloy);
resourceTooltipDict['[[Number.MaxResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.maxResources);

return Localization.instance.getStringWithTokens(_resTooltips[1], resourceTooltipDict);
}

private function getEnergyTooltip():String
{
var resourceTooltipDict:Dictionary = new Dictionary();
resourceTooltipDict['[[Number.BaseResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.energy);
resourceTooltipDict['[[Number.MaxResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.maxResources);

return Localization.instance.getStringWithTokens(_resTooltips[2], resourceTooltipDict);
}

private function getSyntheticTooltip():String
{
var resourceTooltipDict:Dictionary = new Dictionary();
resourceTooltipDict['[[Number.BaseResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.synthetic);
resourceTooltipDict['[[Number.MaxResourceCount]]'] =
StringUtil.commaFormatNumber(_currentBase.maxResources);

return Localization.instance.getStringWithTokens(_resTooltips[3], resourceTooltipDict);
}

private function onMoreClicked( currency:String ):void
{
if (!presenter.hudEnabled)
return;
var storeView:StoreView = StoreView(showView(StoreView));
var filter:uint = StoreView.FILTER_ALLOY;
if (currency == CurrencyEnum.CREDIT)
filter = StoreView.FILTER_CREDITS;
else if (currency == CurrencyEnum.ENERGY)
filter = StoreView.FILTER_ENERGY;
else
```

```
if (currency == CurrencyEnum.SYNTHETIC)
filter = StoreView.FILTER_SYNTHETIC;
storeView.openToResourcesAndFilter(filter);
}

[Inject]
public function set presenter( value:ICommandPresenter ):void { _presenter = value; }
public function get presenter():ICommandPresenter { return CommandPresenter(_presenter); }

override public function get type():String { return ViewEnum.UI; }

override public function destroy():void
{
super.destroy();

tradePresenter.removeTransactionListener(onUpdateTradeRouteTransactions);
tradePresenter = null;

if (_currentBase)
_currentBase.onResourcesChange.remove(onMoneyTick);

_currentBase = null;

tooltip.removeTooltip(null, this);
tooltip = null;

for (var tradeComp:TradeRouteComponent, i:int; i < 3; i++)
{
tradeComp = this["_tradeComp" + i];
tradeComp.destroy();

this["_tradeComp" + i] = null;
}

ObjectPool.give(_resourceComponent);
_resourceComponent = null;
}
}
}

-----------------------------------
File 757: igw\com\ui\hud\shared\command\TradeRouteComponent.as
package com.ui.hud.shared.command
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.model.prototype.IPrototype;
import com.model.starbase.TradeRouteVO;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.ITradePresenter;
import
```

```
com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.ScaleBitmap;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.text.TextFormatAlign;
import flash.utils.Timer;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.osflash.signals.Signal;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class TradeRouteComponent extends Sprite
{
static public const STATE_LOCKED:String = "locked";
static public const STATE_ACTIVE:String = "active";
static public const STATE_RENEW:String = "renew";

public var tradePresenter:ITradePresenter;

public var onClick:Signal;

private var _agentImg:ImageComponent;
private var _renewImg:ScaleBitmap;
private var _lockImg:Bitmap;
private var _frameBtn:BitmapButton;
private var _nextLbl:Label;
private var _timeLbl:Label;
private var _isPendingUpdate:Boolean;
private var _tradeData:TradeRouteVO;
private var _isInitialized:Boolean;
private var _timer:Timer;
private var _unscaledHeight:Number = 0;
private var _unscaledWidth:Number = 0;

private var _nextDeliveryText:String = 'CodeString.TradeRouteComponent.NextDelivery';
//NEXT DELIVERY
private var _expiredText:String = 'CodeString.TradeRouteComponent.Expired'; //EXPIRED
private var _lockedText:String = 'CodeString.TradeRouteComponent.Locked'; //LOCKED

public function init():void
{
_unscaledWidth
```

```
= 60;
_unscaledHeight = 60;

onClick = new Signal(String);

_renewImg = UIFactory.getScaleBitmap("TradeRouteRiverExclamationBMD");
_renewImg.visible = false;
addChild(_renewImg);

_lockImg = UIFactory.getBitmap("IconBlueLockedBMD");
_lockImg.visible = false;
addChild(_lockImg);

_frameBtn = UIFactory.getButton(ButtonEnum.CHARACTER_FRAME);
_frameBtn.addEventListener(MouseEvent.CLICK, onMouseClick);
addChild(_frameBtn);

_agentImg = ObjectPool.get(ImageComponent);
_agentImg.visible = false;
_agentImg.init(54, 54);
addChildAt(_agentImg, 0);

_nextLbl = UIFactory.getLabel(LabelEnum.H5, 100, 25);
_nextLbl.align = TextFormatAlign.CENTER;
_nextLbl.constrictTextToSize = true;
_nextLbl.bold = false;
_nextLbl.textColor = 0xd1e5f7;
_nextLbl.text = "";
addChild(_nextLbl);

_timeLbl = UIFactory.getLabel(LabelEnum.H4, 125, 25);
_timeLbl.align = TextFormatAlign.CENTER;
_timeLbl.constrictTextToSize = false;
_timeLbl.textColor = 0xecffff;
_timeLbl.text = "";
addChild(_timeLbl);

_isInitialized = true;

if (_isPendingUpdate)
update(_tradeData);

_timer = new Timer(1000);
_timer.addEventListener(TimerEvent.TIMER, onTimerTick);
}

protected function updateDisplayList():void
{
if (!_isInitialized)
return;

_frameBtn.setSize(_unscaledWidth,
```

```
_unscaledHeight);

_agentImg.x = _frameBtn.x + (_unscaledWidth - _agentImg.width) / 2;
_agentImg.y = _frameBtn.y + (_unscaledHeight - _agentImg.height) / 2;

_renewImg.x = (_unscaledWidth - _renewImg.width) / 2;
_renewImg.y = (_unscaledHeight - _renewImg.height) / 2;

_lockImg.width = _lockImg.height = 54;
_lockImg.x = (_unscaledWidth - _lockImg.width) / 2;
_lockImg.y = (_unscaledHeight - _lockImg.height) / 2;

switch (_crntState)
{
case STATE_ACTIVE:
{
_agentImg.visible = true;
_renewImg.visible = false;
_lockImg.visible = false;

_nextLbl.text = _nextDeliveryText;
_timeLbl.text = "9m 59s";

TweenLite.killTweensOf(_renewImg);

break;
}

case STATE_RENEW:
{
_agentImg.visible = false;
_renewImg.visible = true;
_lockImg.visible = false;
_nextLbl.text = _expiredText;
_timeLbl.text = "";

onFadeOut();

break;
}

case STATE_LOCKED:
default:
{
_agentImg.visible = false;
_renewImg.visible = false;
_lockImg.visible = true;

_nextLbl.text = _lockedText;
_timeLbl.text = "";

TweenLite.killTweensOf(_renewImg);
```

```
break;
}
}

_nextLbl.x = _frameBtn.x + (_unscaledWidth - _nextLbl.width) / 2;
_nextLbl.y = _frameBtn.y + _frameBtn.height + 2;

_timeLbl.y = _nextLbl.y + _nextLbl.height - 7;

_timer.reset();
_timer.start();
}

private function onMouseClick( event:MouseEvent ):void
{
if (onClick)
onClick.dispatch(_crntState);
}

public function update( tradeData:Object ):void
{
_tradeData = tradeData is TradeRouteVO ? TradeRouteVO(tradeData) : null;

if (!_isInitialized)
{
_isPendingUpdate = true;
return;
}

if (_tradeData)
{
var proto:IPrototype = tradePresenter.getAgent(_tradeData.contractGroup, 1441);
tradePresenter.loadIconFromPrototype("iconImage", proto, onImageLoaded);

_crntState = STATE_ACTIVE;
}

else if (tradeData)
_crntState = STATE_RENEW;

else
_crntState = STATE_LOCKED;

updateDisplayList();
}

private function onTimerTick( e:TimerEvent ):void
{
if
```

```
(!_tradeData)
return;

var freq:int = _tradeData.frequency * 60 * 1000;
var tradeRouteTransaction:TransactionVO =
tradePresenter.getTradeRouteTransaction(_tradeData.id);
if (tradeRouteTransaction)
{
var nextDropMS:int = tradeRouteTransaction.timeRemainingMS % freq;

if (_timeLbl)
{
_timeLbl.setBuildTime(nextDropMS / 1000, 2);
_timeLbl.x = _frameBtn.x + (_unscaledWidth - _timeLbl.width) / 2;
}
}
}

private var _crntState:String = STATE_LOCKED;

private function onImageLoaded( asset:BitmapData ):void
{
if (_agentImg)
_agentImg.onImageLoaded(asset);

updateDisplayList();
}

private function onFadeOut():void
{
TweenLite.to(_renewImg, .5, {alpha:1.0, ease:Quad.easeOut, onComplete:onFadeIn});
}

private function onFadeIn():void
{
TweenLite.to(_renewImg, .5, {alpha:0.0, ease:Quad.easeIn, onComplete:onFadeOut});
}

public function destroy():void
{
_isInitialized = false;

onClick.removeAll();
onClick = null;

tradePresenter = null;

_timer.stop();
_timer.removeEventListener(TimerEvent.TIMER, onTimerTick);
_timer = null;

_frameBtn.removeEventListener(MouseEvent.CLICK,
```

```
onMouseClick);
_frameBtn.destroy();
_frameBtn = null;

_agentImg.destroy();
_agentImg = null;

TweenLite.killTweensOf(_renewImg);
_renewImg.destroy();
_renewImg = null;

_lockImg = null;

_nextLbl.destroy();
_nextLbl = null;

_timeLbl.destroy();
_timeLbl = null;
}
}
}

-----------------------------------
File 758: igw\com\ui\hud\shared\engineering\BuffButton.as
package com.ui.hud.shared.engineering
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.model.asset.AssetVO;
import com.model.starbase.BuffVO;
import com.presenter.shared.IUIPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;

import flash.display.Sprite;

public class BuffButton extends Sprite
{
private var _buff:BuffVO;
private var _button:BitmapButton;
private var _presenter:IUIPresenter;
private var _tempTime:Number;

private var _protectionBuffText:String = 'CodeString.EngineeringView.ProtectionBuff'; //Starbase
Protection

public function init( buff:BuffVO, presenter:IUIPresenter ):void
{
_buff = buff;
_presenter
```

```
= presenter;

//if there is no buff then assume base bubble buff
var buffType:String = (_buff) ? _buff.buffType : "Protection";
var buttonType:String;
switch (buffType)
{
case "Protection":
buttonType = ButtonEnum.BUFF_SHIELD;
break;
case "IncomeAll":
case "IncomeAlloy":
case "IncomeCredits":
case "IncomeEnergy":
case "IncomeSynth":
buttonType = ButtonEnum.BUFF_RESOURCE;
break;
case "BuildSpeed":
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
case "BuildTime":
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
case "MapSpeed":
buttonType = ButtonEnum.BUFF_MAP_SPEED;
break;
case "CargoCapacity":
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
case "DEV_DamageNegation":
case "DEV_IncomeBoost":
case "DEV_BuildAccelerator":
case "DEV_EverythingIsFree":
case "DEV_DisableRequirements":
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
case "DailyBuff":
switch (buff.id)
{
case "Daily_Bounty":
buttonType = ButtonEnum.BUFF_COMMISSION;
break;
case "Daily_RepairSpeed":
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
case "Daily_Salvage":
buttonType = ButtonEnum.BUFF_RESOURCE;
break;
case "Daily_TreasureFinding":
buttonType = ButtonEnum.BUFF_RECLAMATION;
break;
```

```
default:
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
}
break;
case "SalvageBonus":
buttonType = ButtonEnum.BUFF_RESOURCE;
break;
case "BountyBonus":
buttonType = ButtonEnum.BUFF_COMMISSION;
break;
case "RepairSpeed":
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
case "TreasureFinding":
buttonType = ButtonEnum.BUFF_RECLAMATION;
break;
default:
buttonType = ButtonEnum.BUFF_REPAIR_SPEED;
break;
}
_button = UIFactory.getButton(buttonType, 0, 0, 0, 0, "hello", LabelEnum.H4);
_button.label.setSize(35, 35);
_button.label.bold = true;
_button.label.constrictTextToSize = false;
_button.textColor = 0xfef9bd;
_button.label.y = 28;
updateTime();
addChild(_button);
}

public function updateTime():Boolean
{
_tempTime = (_buff) ? _buff.timeRemainingMS : _presenter.bubbleTimeRemaining;
_button.label.setBuildTime(_tempTime * .001, 1);
return _tempTime > 0;
}

public function getTooltip():String
{
if (_buff)
{
var asset:AssetVO = _presenter.getAssetVO(_buff.uiAsset);
if (asset)
return Localization.instance.getString(asset.visibleName);
} else
return Localization.instance.getString(_protectionBuffText);
return "Unknown Buff";
}

public
```

```
function get buff():BuffVO { return _buff; }
public function get buffType():String { return (_buff) ? _buff.buffType : "Protection"; }

public function destroy():void
{
_buff = null;
_button = UIFactory.destroyButton(_button);
_presenter = null;
}
}
}

-----------------------------------
File 759: igw\com\ui\hud\shared\engineering\EngineerGroup.as
package com.ui.hud.shared.engineering
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.TransactionEvent;
import com.model.transaction.TransactionVO;
import com.presenter.shared.IEngineeringPresenter;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.pulldown.DrawerComponent;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class EngineerGroup extends Sprite
{
private var _actionSignal:Signal;
private var _button:BitmapButton;
private var _count:int;
private var _countLabel:Label;
private var _drawer:DrawerComponent;
private var _id:int;
private var _numberBox:Bitmap;
private var _presenter:IEngineeringPresenter;
private var _repairCount:int;
private var _transactionLookup:Dictionary;
private var _trays:Vector.<GroupTray>;

public function init( id:int, title:String, presenter:IEngineeringPresenter ):void
{
```

```
_presenter = presenter;
_actionSignal = new Signal(int, int, int, TransactionVO);
_transactionLookup = new Dictionary(false);
_trays = new Vector.<GroupTray>;

//main button
_button = UIFactory.getButton(ButtonEnum.BLUE_A, 155, 41, 0, 0, title, LabelEnum.H3);
_button.setMargin(36, 0);
_numberBox = UIFactory.getBitmap(PanelEnum.NUMBER_BOX);
_numberBox.x = _numberBox.y = 5;
_countLabel = UIFactory.getLabel(LabelEnum.H3, _numberBox.width, _numberBox.height,
_numberBox.x, _numberBox.y + 1);
_count = _repairCount = 0;
_countLabel.text = _count + "";
_countLabel.textColor = 0xacd1ff;
_button.addChild(_numberBox);
_button.addChild(_countLabel);
_button.addEventListener(MouseEvent.CLICK, onButtonClicked, false, 0, true);

//drawer component
_drawer = ObjectPool.get(DrawerComponent);
_drawer.init(PanelEnum.CONTAINER_DOUBLE_NOTCHED_ARROWS, 152, 115,
DrawerComponent.EXPANDS_DOWN, 5, 30);
_drawer.useMask = true;
_drawer.x = 1.5;
_drawer.y = 40 - _drawer.height;

_id = id;

addChild(_drawer);
addChild(_button);

//set up the default trays based on the id of the group
setupTrays();
}

public function addTransaction( transaction:TransactionVO ):int
{
if (!_transactionLookup.hasOwnProperty(transaction.id + transaction.type))
{
_count++;
_countLabel.text = '' + _count;
_transactionLookup[transaction.id + transaction.type] = transaction;
for (var i:int = 0; i < _trays.length; i++)
_trays[i].addTransaction(transaction);
if (_id == EngineeringView.BUILD && transaction.type ==
TransactionEvent.STARBASE_REPAIR_BASE)
{
_repairCount++;
_drawer.addElement(_trays[0]);
```

```
_trays[1].y = 140;
_drawer.dirty = true;
}
return 1;
}
return 0;
}

public function removeTransaction( transaction:TransactionVO ):int
{
if (_transactionLookup.hasOwnProperty(transaction.id + transaction.type))
{
_count--;
_countLabel.text = '' + _count;
delete _transactionLookup[transaction.id + transaction.type];
for (var i:int = 0; i < _trays.length; i++)
_trays[i].removeTransaction(transaction);
if (_id == EngineeringView.BUILD && transaction.type ==
TransactionEvent.STARBASE_REPAIR_BASE)
{
_repairCount--;
if (_repairCount <= 0)
{
_drawer.removeElement(_trays[0]);
_trays[1].y = 0;
_drawer.dirty = true;
}
}
return 1;
}
return 0;
}

protected function setupTrays():void
{
var tray:GroupTray;
switch (_id)
{
case EngineeringView.BUILD:
//create the repair tray
tray = ObjectPool.get(GroupTray);
tray.init(id, GroupTray.REPAIR_BASE, _actionSignal, _presenter);
_presenter.injectObject(tray);
_trays.push(tray);

//create the build tray
tray = ObjectPool.get(GroupTray);
tray.init(id, GroupTray.STARBASE_BUILD, _actionSignal, _presenter);
_presenter.injectObject(tray);
_trays.push(tray);
```

```
_drawer.addElement(tray);
break;
case EngineeringView.DOCK:
tray = ObjectPool.get(GroupTray);
tray.init(id, GroupTray.REPAIR_SHIP, _actionSignal, _presenter);
_presenter.injectObject(tray);
_trays.push(tray);
_drawer.addElement(tray);
break;
case EngineeringView.RESEARCH:
//weapons / tech tray
tray = ObjectPool.get(GroupTray);
tray.init(id, GroupTray.RESEARCH_A, _actionSignal, _presenter);
_presenter.injectObject(tray);
_trays.push(tray);
_drawer.addElement(tray);

//defense / hulls tray
tray = ObjectPool.get(GroupTray);
tray.init(id, GroupTray.RESEARCH_B, _actionSignal, _presenter);
_presenter.injectObject(tray);
tray.y = 140;
_trays.push(tray);
_drawer.addElement(tray);
break;
case EngineeringView.SHIPYARD:
tray = ObjectPool.get(GroupTray);
tray.init(id, GroupTray.SHIP_BUILD, _actionSignal, _presenter);
_presenter.injectObject(tray);
_trays.push(tray);
_drawer.addElement(tray);
break;
}
}

protected function onButtonClicked( e:MouseEvent ):void
{
if (!_presenter.hudEnabled)
return;
_actionSignal.dispatch(_id, -1, -1, null);
e.stopPropagation();
}

public function updateTransactionTime():void
{
for (var i:int = 0; i < _trays.length; i++)
_trays[i].updateTransactionTime();
}

public
```

```
function addActionListener( listener:Function ):void { _actionSignal.add(listener); }
public function removeActionListener( listener:Function ):void { _actionSignal.remove(listener); }

public function set buttonText( v:String ):void { _button.text = v; }
public function set canDragExpand( v:Boolean ):void { _drawer.canDragExpand = v; }
public function get id():int { return _id; }

public function destroy():void
{
_actionSignal.removeAll();
_actionSignal = null;

_button.removeEventListener(MouseEvent.CLICK, onButtonClicked);
_button = UIFactory.destroyButton(_button);
_countLabel = UIFactory.destroyLabel(_countLabel);
ObjectPool.give(_drawer);
_drawer = null;
_numberBox = UIFactory.destroyPanel(_numberBox);
_presenter = null;
_transactionLookup = null;

for (var i:int = 0; i < _trays.length; i++)
_trays[i].destroy();
_trays.length = 0;
}
}
}

-----------------------------------
File 760: igw\com\ui\hud\shared\engineering\EngineeringView.as
package com.ui.hud.shared.engineering
{
import com.Application;
import com.enum.PositionEnum;
import com.enum.StarbaseCategoryEnum;
import com.enum.TypeEnum;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.event.TransactionEvent;
import com.event.signal.TransactionSignal;
import com.model.transaction.TransactionVO;
import com.presenter.shared.IEngineeringPresenter;
import com.ui.core.View;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.core.effects.EffectFactory;
import com.ui.modal.construction.ConstructionInfoView;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.dock.DockView;
import com.ui.modal.shipyard.ShipyardView;
import com.ui.modal.store.StoreView;

import
```

```
flash.events.Event;
import flash.events.TimerEvent;
import flash.geom.Rectangle;
import flash.utils.Dictionary;
import flash.utils.Timer;

import org.parade.core.IView;
import org.parade.enum.ViewEnum;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class EngineeringView extends View
{
public static const BUILD:int = 0;
public static const DOCK:int = 1;
public static const RESEARCH:int = 2;
public static const SHIPYARD:int = 3;

private var _buildGroup:EngineerGroup;
private var _count:int;
private var _dockGroup:EngineerGroup;
private var _researchGroup:EngineerGroup;
private var _shipyardGroup:EngineerGroup;
private var _timer:Timer;
private var _tooltip:Tooltips;

private const MIN_X_POS:Number = 385;

private var _buffDescription:String = 'CodeString.EngineeringView.BuffInfo'; //Click here to get
more buffs
private var _buildText:String = 'CodeString.Controls.Build'; //BUILD
private var _fleetsText:String = 'CodeString.Controls.Fleets'; //FLEETS
private var _researchText:String = 'CodeString.Controls.Research'; //RESEARCH
private var _shipyardText:String = 'CodeString.Controls.Shipyard'; //SHIPYARD

[PostConstruct]
override public function init():void
{
super.init();
presenter.addTransactionListener(TransactionSignal.TRANSACTION_UPDATED,
onTransactionUpdated);
presenter.addTransactionListener(TransactionSignal.TRANSACTION_REMOVED,
onTransactionRemoved);
y = 4;
_count = 0;
//initialize groups
_buildGroup = ObjectPool.get(EngineerGroup);
_buildGroup.init(BUILD, _buildText, presenter);
_buildGroup.addActionListener(onGroupAction);
_buildGroup.canDragExpand = !presenter.inFTE;

_dockGroup
```

```
= ObjectPool.get(EngineerGroup);
_dockGroup.init(DOCK, _fleetsText, presenter);
_dockGroup.addActionListener(onGroupAction);
_dockGroup.canDragExpand = !presenter.inFTE;

_researchGroup = ObjectPool.get(EngineerGroup);
_researchGroup.init(RESEARCH, _researchText, presenter);
_researchGroup.addActionListener(onGroupAction);
_researchGroup.canDragExpand = !presenter.inFTE;

_shipyardGroup = ObjectPool.get(EngineerGroup);
_shipyardGroup.init(SHIPYARD, _shipyardText, presenter);
_shipyardGroup.addActionListener(onGroupAction);
_shipyardGroup.canDragExpand = !presenter.inFTE;

//position groups
_researchGroup.x = _buildGroup.x + _buildGroup.width + 6;
_shipyardGroup.x = _researchGroup.x + _researchGroup.width + 6;
_dockGroup.x = _shipyardGroup.x + _shipyardGroup.width + 6;

//transaction update timer
_timer = new Timer(1000);
addListener(_timer, TimerEvent.TIMER, onTimer);

//add to the display list
addChild(_buildGroup);
addChild(_dockGroup);
addChild(_researchGroup);
addChild(_shipyardGroup);

//initialize with the existing transactions
var transactions:Dictionary = presenter.transactions;
for each (var transaction:TransactionVO in transactions)
onTransactionUpdated(transaction);

//show the view
onStageResized();
addHitArea();
addEffects();
effectsIN();

visible = !presenter.inFTE;
}

private function onTransactionUpdated( transaction:TransactionVO ):void
{
var group:EngineerGroup = getGroupForTransaction(transaction);
if (group)
{
_count += group.addTransaction(transaction);
if
```

```
(!_timer.running && _count > 0)
_timer.start();
}
}

private function onTransactionRemoved( transaction:TransactionVO ):void
{
var group:EngineerGroup = getGroupForTransaction(transaction);
if (group)
{
_count -= group.removeTransaction(transaction);
if (_count == 0)
_timer.reset();
}
}

private function getGroupForTransaction( transaction:TransactionVO ):EngineerGroup
{
if (transaction.timeMS <= 0)
return null;
switch (transaction.type)
{
case TransactionEvent.STARBASE_BUILD_SHIP:
case TransactionEvent.STARBASE_REFIT_SHIP:
return _shipyardGroup;

case TransactionEvent.STARBASE_BUILDING_BUILD:
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
case TransactionEvent.STARBASE_REFIT_BUILDING:
case TransactionEvent.STARBASE_REPAIR_BASE:
return _buildGroup;

case TransactionEvent.STARBASE_REPAIR_FLEET:
return _dockGroup;

case TransactionEvent.STARBASE_RESEARCH:
return _researchGroup;
}

return null;
}

private function onGroupAction( id:int, type:int, index:int, transaction:TransactionVO ):void
{
if (!presenter.hudEnabled)
return;
if (transaction)
{
var storeView:StoreView = StoreView(showView(StoreView));
storeView.setSelectedTransaction(transaction);
}
```

```
else
{
var hasBuilding:Boolean;
var view:IView;
switch (id)
{
case BUILD:
switch (type)
{
case GroupTray.STARBASE_BUILD:
if (Application.STATE == StateEvent.GAME_STARBASE)
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.BUILD, index == 0 ?
StarbaseCategoryEnum.INFRASTRUCTURE : StarbaseCategoryEnum.DEFENSE, null);
_viewFactory.notify(view);
} else
enterStarbase(ConstructionView, {type:ConstructionView.BUILD, groupID:(index == 0 ?
StarbaseCategoryEnum.INFRASTRUCTURE : StarbaseCategoryEnum.DEFENSE)});
break;
default:
if (Application.STATE == StateEvent.GAME_STARBASE)
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.BUILD, null, null);
_viewFactory.notify(view);
} else
enterStarbase(ConstructionView, {type:ConstructionView.BUILD,
groupID:StarbaseCategoryEnum.INFRASTRUCTURE});
break;
}
break;
case DOCK:
showView(DockView);
break;
case RESEARCH:
switch (type)
{
case GroupTray.RESEARCH_A:
hasBuilding = presenter.getBuildingCount(index == 0 ? TypeEnum.WEAPONS_FACILITY :
TypeEnum.ADVANCED_TECH) > 0;
if (Application.STATE == StateEvent.GAME_STARBASE)
{
if (!hasBuilding)
{
view = _viewFactory.createView(ConstructionInfoView);
ConstructionInfoView(view).setup(ConstructionView.BUILD,
presenter.getBuildingPrototypeByClassAndLevel(index == 0 ?
TypeEnum.WEAPONS_FACILITY : TypeEnum.ADVANCED_TECH, 1));
_viewFactory.notify(view);
}
```

```
else
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH, index == 0 ?
TypeEnum.WEAPONS_FACILITY : TypeEnum.ADVANCED_TECH, null);
_viewFactory.notify(view);
}
} else
{
if (!hasBuilding)
enterStarbase(ConstructionInfoView, {type:ConstructionView.BUILD,
proto:presenter.getBuildingPrototypeByClassAndLevel(index == 0 ?
TypeEnum.WEAPONS_FACILITY : TypeEnum.ADVANCED_TECH, 1)});
else
enterStarbase(ConstructionView, {type:ConstructionView.RESEARCH, groupID:(index == 0 ?
TypeEnum.WEAPONS_FACILITY : TypeEnum.ADVANCED_TECH)});
}
break;
case GroupTray.RESEARCH_B:
hasBuilding = presenter.getBuildingCount(index == 0 ? TypeEnum.DEFENSE_DESIGN :
TypeEnum.SHIPYARD) > 0;
if (Application.STATE == StateEvent.GAME_STARBASE)
{
if (!hasBuilding)
{
view = _viewFactory.createView(ConstructionInfoView);
ConstructionInfoView(view).setup(ConstructionView.BUILD,
presenter.getBuildingPrototypeByClassAndLevel(index == 0 ? TypeEnum.DEFENSE_DESIGN :
TypeEnum.SHIPYARD, 1));
_viewFactory.notify(view);
} else
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH, index == 0 ?
TypeEnum.DEFENSE_DESIGN : TypeEnum.SHIPYARD, null);
_viewFactory.notify(view);
}
} else
{
if (!hasBuilding)
enterStarbase(ConstructionInfoView, {type:ConstructionView.BUILD,
proto:presenter.getBuildingPrototypeByClassAndLevel(index == 0 ?
TypeEnum.DEFENSE_DESIGN : TypeEnum.SHIPYARD, 1)});
else
enterStarbase(ConstructionView, {type:ConstructionView.RESEARCH, groupID:(index == 0 ?
TypeEnum.DEFENSE_DESIGN : TypeEnum.SHIPYARD)});
}
break;
default:
if (Application.STATE == StateEvent.GAME_STARBASE)
{
```

```
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH, null, null);
_viewFactory.notify(view);
} else
enterStarbase(ConstructionView, {type:ConstructionView.RESEARCH});
break;
}
break;
case SHIPYARD:
if (Application.STATE == StateEvent.GAME_STARBASE)
showView(ShipyardView);
else
enterStarbase(ShipyardView, null);
break;
}
}
}

override protected function onStateChange( state:String ):void
{
if (state == StateEvent.GAME_BATTLE)
destroy();
}

private function onTimer( e:TimerEvent ):void
{
_buildGroup.updateTransactionTime();
_dockGroup.updateTransactionTime();
_researchGroup.updateTransactionTime();
_shipyardGroup.updateTransactionTime();
}

private function enterStarbase( view:Class, viewData:* ):void
{
var starbaseEvent:StarbaseEvent = new StarbaseEvent(StarbaseEvent.ENTER_BASE);
starbaseEvent.view = view;
starbaseEvent.viewData = viewData;
presenter.dispatch(starbaseEvent);
}

override protected function addEffects():void {
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.CENTER, PositionEnum.TOP,
onStageResized)); }

private function onStageResized( e:Event = null ):void
{
this.scaleX = this.scaleY = Application.SCALE;
var bounds:Rectangle = this.getBounds(this);
var pos:Number = MIN_X_POS * Application.SCALE;
x
```

```
= (((DeviceMetrics.WIDTH_PIXELS - super.width) * .5 - bounds.x) > pos ?
((DeviceMetrics.WIDTH_PIXELS - super.width) * .5 - bounds.x) : pos);
}

override public function get height():Number { return super.height * Application.SCALE; }
override public function get width():Number { return super.width * Application.SCALE; }

[Inject]
public function set presenter( value:IEngineeringPresenter ):void { _presenter = value; }
public function get presenter():IEngineeringPresenter { return
IEngineeringPresenter(_presenter); }

[Inject]
public function set tooltip( value:Tooltips ):void { _tooltip = value; }

override public function get type():String { return ViewEnum.UI }
override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
presenter.removeTransactionListener(onTransactionUpdated);
presenter.removeTransactionListener(onTransactionRemoved);

super.destroy();

ObjectPool.give(_buildGroup);
_buildGroup = null;
ObjectPool.give(_dockGroup);
_dockGroup = null;
ObjectPool.give(_researchGroup);
_researchGroup = null;
ObjectPool.give(_shipyardGroup);
_shipyardGroup = null;

_timer.reset();
_timer = null;

_tooltip.removeTooltip(null, this);
_tooltip = null;
}
}
}

-----------------------------------
File 761: igw\com\ui\hud\shared\engineering\GroupButton.as
package com.ui.hud.shared.engineering
{
import com.enum.TypeEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import
```

```
com.model.transaction.TransactionVO;
import com.presenter.shared.IEngineeringPresenter;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Sprite;
import flash.events.MouseEvent;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class GroupButton extends Sprite
{
private var _actionSignal:Signal;
private var _bottomLabel:Label;
private var _button:BitmapButton;
private var _image:ImageComponent;
private var _index:int;
private var _middleLabel:Label;
private var _parentID:int;
private var _presenter:IEngineeringPresenter;
private var _topLabel:Label;
private var _transaction:TransactionVO;
private var _type:int;

private var _speedUpText:String = 'CodeString.Shared.SpeedUp'; //SPEED UP
private var _noRepairText:String = 'CodeString.EngineeringView.NoRepair'; //NO REPAIR
private var _noResearchText:String = 'CodeString.EngineeringView.NoResearch'; //NO
RESEARCH
private var _noBuildText:String = 'CodeString.EngineeringView.NoBuild'; //NO BUILD
private var _noDefenseText:String = 'CodeString.EngineeringView.NoDefense'; //NO DEFENSE

public function init( parentID:int, type:int, index:int, actionSignal:Signal,
presenter:IEngineeringPresenter ):void
{
_actionSignal = actionSignal;
_index = index;
_parentID = parentID;
_presenter = presenter;
_type = type;

_bottomLabel = UIFactory.getLabel(LabelEnum.H4, 60, 28, 0, 57);
_bottomLabel.textColor = 0xffe3b2;
_bottomLabel.constrictTextToSize = true;
_bottomLabel.text = "";

_image = ObjectPool.get(ImageComponent);
_image.init(60, 60);
_image.center
```

```
= _image.smoothing = true;

_middleLabel = UIFactory.getLabel(LabelEnum.DEFAULT, 60, 60, 0, 0);
_middleLabel.multiline = _middleLabel.bold = _middleLabel.constrictTextToSize = true;
_middleLabel.textColor = 0x597181;
_middleLabel.text = '';

_topLabel = UIFactory.getLabel(LabelEnum.DEFAULT, 60, 30, 0, -18);
_topLabel.fontSize = 15;
_topLabel.text = "";

showState();
}

public function addTransaction( transaction:TransactionVO ):void
{
_transaction = transaction;
_presenter.loadTransactionIcon(transaction, _image.onImageLoaded);
_bottomLabel.text = _speedUpText;
_topLabel.setBuildTime(transaction.timeRemainingMS * .001, 2);
showState();
}

public function updateTransactionTime():void
{
if (_transaction)
_topLabel.setBuildTime(transaction.timeRemainingMS * .001, 2);
}

public function removeTransaction( transaction:TransactionVO ):void
{
_transaction = null;
_image.clearBitmap();
showState();
}

public function showState():void
{
if (!_transaction)
{
_bottomLabel.text = '';
_topLabel.text = '';
}

var hasBuilding:Boolean;
var middleText:String = '';
switch (_type)
{
case GroupTray.REPAIR_BASE:
middleText = _noRepairText;
skin
```

```
= ButtonEnum.FRAME_RED;
break;
case GroupTray.REPAIR_SHIP:
middleText = _noRepairText;
skin = ButtonEnum.FRAME_RED;
break;
case GroupTray.RESEARCH_A:
middleText = _noResearchText;
if (!_transaction)
{
hasBuilding = _presenter.getBuildingCount(_index == 0 ? TypeEnum.WEAPONS_FACILITY :
TypeEnum.ADVANCED_TECH) > 0;
skin = (true) ? ButtonEnum.FRAME_BLUE : ButtonEnum.FRAME_RED;
} else
skin = ButtonEnum.FRAME_BLUE;
break;
case GroupTray.RESEARCH_B:
middleText = _noResearchText;
if (!_transaction)
{
hasBuilding = _presenter.getBuildingCount(_index == 0 ? TypeEnum.DEFENSE_DESIGN :
TypeEnum.SHIPYARD) > 0;
skin = (true) ? ButtonEnum.FRAME_BLUE : ButtonEnum.FRAME_RED;
} else
skin = ButtonEnum.FRAME_BLUE;
break;
case GroupTray.SHIP_BUILD:
middleText = _noBuildText;
skin = ButtonEnum.FRAME_BLUE;
break;
case GroupTray.STARBASE_BUILD:
middleText = (_index == 0) ? _noBuildText : _noDefenseText;
skin = ButtonEnum.FRAME_BLUE;
break;
}
_middleLabel.htmlText = (_transaction) ? '' : middleText;
_middleLabel.y = (60 - _middleLabel.textHeight) * .5 - 4;
}

protected function onMouseClick( e:MouseEvent ):void { _actionSignal.dispatch(_parentID,
_type, _index, _transaction); e.stopPropagation(); }

public function get image():ImageComponent { return _image; }

private function set skin( type:String ):void
{
if (_button)
{
_button.removeEventListener(MouseEvent.CLICK, onMouseClick);
removeChild(_button);
_button
```

```
= UIFactory.destroyButton(_button);
}
if (type)
{
_button = UIFactory.getButton(type, 60, 60, 0, 20);
_button.addChild(_bottomLabel);
_button.addChild(_image);
_button.addChild(_middleLabel);
_button.addChild(_topLabel);
_button.addEventListener(MouseEvent.CLICK, onMouseClick, false, 0, true);
addChild(_button);
}
}

public function get transaction():TransactionVO { return _transaction; }

public function destroy():void
{
_actionSignal = null;
_presenter = null;
skin = null;
}
}
}

------------------------------------
File 762: igw\com\ui\hud\shared\engineering\GroupTray.as
package com.ui.hud.shared.engineering
{
import com.enum.StarbaseConstructionEnum;
import com.enum.TypeEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.TransactionEvent;
import com.model.asset.AssetVO;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.starbase.BuildingVO;
import com.model.starbase.ResearchVO;
import com.model.transaction.TransactionVO;
import com.presenter.shared.IEngineeringPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;

import flash.display.Sprite;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

import
```

```
org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class GroupTray extends Sprite
{
public static const REPAIR_BASE:int = 0;
public static const REPAIR_SHIP:int = 1;
public static const RESEARCH_A:int = 2;
public static const RESEARCH_B:int = 3;
public static const SHIP_BUILD:int = 4;
public static const STARBASE_BUILD:int = 5;

private var _actionSignal:Signal;
private var _bg:ScaleBitmap;
private var _buttonLeft:GroupButton;
private var _buttonRight:GroupButton;
private var _numButtons:int;
private var _parentID:int;
private var _presenter:IEngineeringPresenter;
private var _titleLeft:Label;
private var _titleRight:Label;
private var _type:int;
private var _tooltip:Tooltips;

private var _repairText:String = 'CodeString.EngineeringView.Repair' //REPAIR
private var _weaponsText:String = 'CodeString.EngineeringView.Weapons' //WEAPONS
private var _techText:String = 'CodeString.EngineeringView.Tech' //TECH
private var _defenseText:String = 'CodeString.EngineeringView.Defense' //DEFENSE
private var _hullsText:String = 'CodeString.EngineeringView.Hulls' //HULLS
private var _buildText:String = 'CodeString.EngineeringView.Build' //BUILD

public function init( parentID:int, type:int, actionSignal:Signal, presenter:IEngineeringPresenter
):void
{
_actionSignal = actionSignal;
_parentID = parentID;
_presenter = presenter;
_type = type;

var panelType:String;
switch (_type)
{
case REPAIR_BASE:
titleLeft = _repairText;
_numButtons = 2;
panelType = PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL;
break;
case REPAIR_SHIP:
titleLeft = _repairText;
_numButtons = 1;
panelType
```

```
= PanelEnum.CONTAINER_DOUBLE_NOTCHED;
break;
case RESEARCH_A:
titleLeft = _weaponsText;
titleRight = _techText;
_numButtons = 2;
panelType = PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL;
break;
case RESEARCH_B:
titleLeft = _defenseText;
titleRight = _hullsText;
_numButtons = 2;
panelType = PanelEnum.CONTAINER_DOUBLE_NOTCHED;
break;
case SHIP_BUILD:
titleLeft = _buildText;
_numButtons = 1;
panelType = PanelEnum.CONTAINER_DOUBLE_NOTCHED;
break;
case STARBASE_BUILD:
titleLeft = _buildText;
_numButtons = 2;
panelType = PanelEnum.CONTAINER_DOUBLE_NOTCHED;
break;
}

_bg = UIFactory.getPanel(panelType, 143, 111, 4, 24);
addChild(_bg);
if (_numButtons > 0)
{
_buttonLeft = ObjectPool.get(GroupButton);
_buttonLeft.init(_parentID, _type, 0, _actionSignal, _presenter);
_buttonLeft.x = 12;
_buttonLeft.y = 28;
addChild(_buttonLeft);
}
if (_numButtons > 1)
{
_buttonRight = ObjectPool.get(GroupButton);
_buttonRight.init(_parentID, _type, 1, _actionSignal, _presenter);
_buttonRight.x = 78;
_buttonRight.y = 28;
addChild(_buttonRight);
}
}

public function addTransaction( transaction:TransactionVO ):void
{
var research:ResearchVO;
var assetVO:AssetVO;
switch
```

```
(_type)
{
case RESEARCH_A:
research = _presenter.getResearchByID(transaction.id);

if (research)
{
assetVO = _presenter.getAssetVO(research.uiAsset);

if (research.requiredBuildingClass == TypeEnum.WEAPONS_FACILITY)
{
_buttonLeft.addTransaction(transaction);
_tooltip.addTooltip(_buttonLeft, this, null,
Localization.instance.getString(assetVO.visibleName));
} else if (research.requiredBuildingClass == TypeEnum.ADVANCED_TECH)
{
_buttonRight.addTransaction(transaction);
_tooltip.addTooltip(_buttonRight, this, null,
Localization.instance.getString(assetVO.visibleName));
}
}
break;
case RESEARCH_B:
research = _presenter.getResearchByID(transaction.id);
if (research)
{
assetVO = _presenter.getAssetVO(research.uiAsset);
if (research.requiredBuildingClass == TypeEnum.DEFENSE_DESIGN)
{
_buttonLeft.addTransaction(transaction);
_tooltip.addTooltip(_buttonLeft, this, null,
Localization.instance.getString(assetVO.visibleName));
} else if (research.requiredBuildingClass == TypeEnum.SHIPYARD)
{
_buttonRight.addTransaction(transaction);
_tooltip.addTooltip(_buttonRight, this, null,
Localization.instance.getString(assetVO.visibleName));
}
}
break;
case STARBASE_BUILD:
//if (transaction.type == TransactionEvent.STARBASE_REPAIR_BASE)
// return;
var building:BuildingVO = _presenter.getBuildingByID(transaction.id);
if (building)
{
assetVO = _presenter.getAssetVO(building.uiAsset);
if (building.constructionCategory == StarbaseConstructionEnum.DEFENSE)
{
_buttonRight.addTransaction(transaction);
_tooltip.addTooltip(_buttonRight,
```

```
this, null, Localization.instance.getString(assetVO.visibleName));
} else
{
_buttonLeft.addTransaction(transaction);
_tooltip.addTooltip(_buttonLeft, this, null,
Localization.instance.getString(assetVO.visibleName));
}
}
break;

case SHIP_BUILD:
var ship:ShipVO = _presenter.getShipVOByID(transaction.id);

if (ship)
{
assetVO = _presenter.getAssetVO(ship.uiAsset);
if (_numButtons > 0 && _buttonLeft.transaction == null)
{
_buttonLeft.addTransaction(transaction);
_tooltip.addTooltip(_buttonLeft, this, null,
Localization.instance.getString(assetVO.visibleName));
} else if (_numButtons > 1 && _buttonRight.transaction == null)
{
_buttonRight.addTransaction(transaction);
_tooltip.addTooltip(_buttonRight, this, null,
Localization.instance.getString(assetVO.visibleName));
}
}
break;
case REPAIR_BASE:
if (transaction.type != TransactionEvent.STARBASE_REPAIR_BASE)
return;
var repairBuilding:BuildingVO = _presenter.getBuildingByID(transaction.id);

if (building)
{
assetVO = _presenter.getAssetVO(repairBuilding.uiAsset);
if (_numButtons > 0 && _buttonLeft.transaction == null)
{
_buttonLeft.addTransaction(transaction);
_tooltip.addTooltip(_buttonLeft, this, null,
Localization.instance.getString(assetVO.visibleName));
} else if (_numButtons > 1 && _buttonRight.transaction == null)
{
_buttonLeft.addTransaction(transaction);
_tooltip.addTooltip(_buttonRight, this, null,
Localization.instance.getString(assetVO.visibleName));
}
}
break;
case
```

```
REPAIR_SHIP:
var fleet:FleetVO = _presenter.getRepairFleetByID(transaction.id);

if (fleet)
{
if (_numButtons > 0 && _buttonLeft.transaction == null)
_tooltip.addTooltip(_buttonLeft, this, null, fleet.name);
else if (_numButtons > 1 && _buttonRight.transaction == null)
_tooltip.addTooltip(_buttonRight, this, null, fleet.name);
}


default:
if (_numButtons > 0 && _buttonLeft.transaction == null)
_buttonLeft.addTransaction(transaction);
else if (_numButtons > 1 && _buttonRight.transaction == null)
_buttonRight.addTransaction(transaction);
break;

}
}

public function removeTransaction( transaction:TransactionVO ):void
{
if (_numButtons > 0 && _buttonLeft.transaction == transaction)
_buttonLeft.removeTransaction(transaction);
else if (_numButtons > 1 && _buttonRight.transaction == transaction)
_buttonRight.removeTransaction(transaction);
if (_type == REPAIR_BASE)
{
//find a new repair transaction
var transactions:Dictionary = _presenter.transactions;
for each (var trans:TransactionVO in transactions)
{
if (trans.type == TransactionEvent.STARBASE_REPAIR_BASE)
{
if (_buttonLeft.transaction == null && _buttonRight.transaction != trans)
_buttonLeft.addTransaction(trans);
else if (_buttonRight.transaction == null && _buttonLeft.transaction != trans)
_buttonRight.addTransaction(trans);
}
}
}
}

public function updateTransactionTime():void
{
if (_buttonLeft)
_buttonLeft.updateTransactionTime();
if (_buttonRight)
_buttonRight.updateTransactionTime();
```

```
}

protected function set titleLeft( v:String ):void
{
if (v)
{
if (!_titleLeft)
{
_titleLeft = UIFactory.getLabel(LabelEnum.SUBTITLE, 70, 30, 3, 0);
_titleLeft.align = TextFormatAlign.LEFT;
_titleLeft.bold = false;
_titleLeft.constrictTextToSize = true;
}
_titleLeft.text = v;
addChild(_titleLeft);
} else if (_titleLeft)
{
removeChild(_titleLeft);
_titleLeft = UIFactory.destroyLabel(_titleLeft);
}
}

protected function set titleRight( v:String ):void
{
if (v)
{
if (!_titleRight)
{
_titleRight = UIFactory.getLabel(LabelEnum.SUBTITLE, 70, 30, 74, 0);
_titleRight.align = TextFormatAlign.LEFT;
_titleRight.bold = false;
_titleRight.constrictTextToSize = true;
}
_titleRight.text = v;
addChild(_titleRight);
} else if (_titleRight)
{
removeChild(_titleRight);
_titleRight = UIFactory.destroyLabel(_titleRight);
}
}

[Inject]
public function set tooltip( value:Tooltips ):void { _tooltip = value; }

public function destroy():void
{
_actionSignal = null;
_presenter = null;
titleLeft
```

```
= null;
titleRight = null;
x = y = 0;
_tooltip.removeTooltip(null, this);
_tooltip = null;
}
}
}

------------------------------------
File 763: igw\com\ui\hud\shared\leaderboards\LeaderboardEntry.as
package com.ui.hud.shared.leaderboards
{
import com.enum.LeaderboardEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.leaderboards.LeaderboardEntryVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.presenter.shared.ILeaderboardPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Point;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;
import org.adobe.utils.StringUtil;

public class LeaderboardEntry extends Sprite
{
public var onClick:Signal;

private var _presenter:ILeaderboardPresenter;

private var _bg:ScaleBitmap;

private var _entryOne:Label;
private var _entryTwo:Label;
private
```

```
var _entryThree:Label;
private var _entryFour:Label;
private var _entryFive:Label;

private var _tooltipHitArea:Sprite;

private var _image:ImageComponent;

private var _getSector:Function;

private var _data:LeaderboardEntryVO;

private var _scope:int;
private var _type:int;
private var _rank:int;

private var _width:int;

private var _isCurrentUser:Boolean;

private var HEIGHT:Number = 41;

private var _tooltip:Tooltips;

private var _noneText:String = 'CodeString.Shared.None'; //None
private var _winLossTooltipText:String = 'CodeString.Leaderboard.Tooltip.WinLoss'; //Kills:
[[Number.Kills]] \nLosses: [[Number.Losses]]
private var _commendationTooltipText:String =
'CodeString.Leaderboard.Tooltip.CommendationRank'; //Commendation Points PvE:
[[Number.PvEPoints]] \nCommendation Points PvP: [[Number.PvPPoints]]


public function LeaderboardEntry( presenter:ILeaderboardPresenter )
{
_presenter = presenter;

onClick = new Signal(String);

var center:Number = HEIGHT * 0.5 - 9;

_entryOne = new Label(16, 0xf0f0f0, 100, 39, false);
_entryOne.constrictTextToSize = false;
_entryOne.align = TextFormatAlign.CENTER;
_entryOne.y = center;

_entryTwo = new Label(16, 0xf0f0f0, 100, 39, false);
_entryTwo.constrictTextToSize = false;
_entryTwo.align = TextFormatAlign.CENTER;
_entryTwo.y = center;

_entryThree
```

```
= new Label(16, 0xf0f0f0, 100, 39, false);
_entryThree.constrictTextToSize = false;
_entryThree.align = TextFormatAlign.CENTER;
_entryThree.y = center;

_entryFour = new Label(16, 0xf0f0f0, 100, 39, false);
_entryFour.constrictTextToSize = false;
_entryFour.align = TextFormatAlign.CENTER;
_entryFour.y = center;

_entryFive = new Label(16, 0xf0f0f0, 100, 39);
_entryFive.constrictTextToSize = false;
_entryFive.align = TextFormatAlign.CENTER;
_entryFive.y = center;

_image = ObjectPool.get(ImageComponent);
_image.init(34, 34);
_image.x = 375;
_image.y = 3;

_tooltipHitArea = new Sprite();

addChild(_entryOne);
addChild(_entryTwo);
addChild(_entryThree);
addChild(_entryFour);
addChild(_entryFive);
addChild(_image);
addChild(_tooltipHitArea);
}

public function update( width:int, lastTypeXPos:int, scope:int, type:int,
data:LeaderboardEntryVO, factionColor:uint, rankAssetVO:AssetVO, isCurrentUser:Boolean
):void
{
_tooltip.removeTooltip(_tooltipHitArea);
_tooltipHitArea.x = lastTypeXPos;
_tooltipHitArea.graphics.clear();
_tooltipHitArea.graphics.beginFill(0xf0f0f0, 0.0);
_tooltipHitArea.graphics.drawRect(0, 0, (width - lastTypeXPos), 40);
_tooltipHitArea.graphics.endFill();

var filteredLabel:Label;
var offset:int;
var localization:Localization;
var tooltip:String = '';
_data = data;
_scope = scope;
_type = type;
_width = width;
_isCurrentUser
```

```
= isCurrentUser;

_entryTwo.text = _data.name;
_entryTwo.textColor = factionColor;

_entryTwo.width = _entryTwo.textWidth + 5;

_image.visible = false;
if (data.isAlliance)
{
buttonMode = false;
removeEventListener(MouseEvent.CLICK, onShowProfileClick);
_entryFour.useLocalization = false;
_entryFive.visible = false;
if (_type == LeaderboardEnum.ALLIANCE_NUM_MEMBERS)
{
filteredLabel = _entryThree;
_entryThree.textColor = factionColor;
_entryFour.visible = false;
} else
{
_entryFour.visible = true;
_entryFour.textColor = factionColor;
filteredLabel = _entryFour;

_entryThree.useLocalization = false;
_entryThree.text = String(_data.numOfMembers);
_entryThree.textColor = factionColor;
_entryThree.width = _entryThree.textWidth + 5;
_entryThree.x = 275 + (150 - _entryThree.width) * 0.5;
}
_entryTwo.x = 51 + (223 - _entryTwo.width) * 0.5;

} else
{
buttonMode = true;
addEventListener(MouseEvent.CLICK, onShowProfileClick, false, 0, true);
filteredLabel = _entryFive;
_entryFive.textColor = factionColor;
_entryFive.visible = true;
_entryFour.visible = true;

if (_data.allianceName != '')
{
_entryThree.useLocalization = false;
_entryThree.text = _data.allianceName;
} else
{
_entryThree.useLocalization = true;
_entryThree.text = _noneText;
}
```

```
_entryTwo.x = 51 + (118 - _entryTwo.width) * 0.5;

_entryThree.textColor = factionColor;
_entryThree.width = _entryThree.textWidth + 5;
_entryThree.x = 170 + (223 - _entryThree.width) * 0.5;

_entryFour.useLocalization = true;
_entryFour.text = _presenter.getSectorName(_data.sectorOwner);
_entryFour.textColor = factionColor;
_entryFour.width = _entryFour.textWidth + 5;
_entryFour.x = 394 + (126 - _entryFour.width) * 0.5;
}

switch (_type)
{
case LeaderboardEnum.ALLIANCE_NUM_MEMBERS:
_rank = _data.numOfMembersRank;
filteredLabel.text = StringUtil.commaFormatNumber(Math.round(_data.numOfMembers));
break;
case LeaderboardEnum.BASE_RATING:
_rank = _data.baseRatingRank;
filteredLabel.text = StringUtil.commaFormatNumber(Math.round(_data.baseRating));
break;
case LeaderboardEnum.ALLIANCE_TOTAL_RATING:
_rank = _data.highestBaseRanking;
filteredLabel.text = StringUtil.commaFormatNumber(Math.round(_data.highestBaseRating));
break;
case LeaderboardEnum.EXPERIENCE:
_rank = _data.experienceRank;
if (data.isAlliance)
filteredLabel.text = StringUtil.commaFormatNumber(_data.experience);
else
filteredLabel.text =
StringUtil.commaFormatNumber(CommonFunctionUtil.findPlayerLevel(_data.experience));
break;
case LeaderboardEnum.COMMENDATION_COMBINED:
_rank = _data.commendiationCombinedRank;
localization = Localization.instance;
if (data.isAlliance)
{
filteredLabel.text = StringUtil.commaFormatNumber(_data.totalCommendationScore);
} else
{
_rank = _data.commendiationCombinedRank;
filteredLabel.useLocalization = true;
filteredLabel.text = rankAssetVO.visibleName;
_presenter.loadIcon(rankAssetVO.iconImage, _image.onImageLoaded);
_image.filters =
[CommonFunctionUtil.getColorMatrixFilter(CommonFunctionUtil.getRankColorBasedOnScore(_data.comm
```

```
_data.commendationPointsPvP))];
_image.x = lastTypeXPos + 5;
_image.visible = true;

if (width - lastTypeXPos != 292)
offset = 10;
}
tooltip = localization.getStringWithTokens(_commendationTooltipText,
{"[[Number.PvEPoints]]":data.commendationPointsPvE,
"[[Number.PvPPoints]]":data.commendationPointsPvP});
break;
case LeaderboardEnum.HIGHEST_FLEET_RATING:
_rank = _data.highestFleetRank;
filteredLabel.text = StringUtil.commaFormatNumber(data.highestFleetRating);
break;
case LeaderboardEnum.ALLIANCE_AVERAGE_HIGHEST_FLEET_RATING:
_rank = _data.avgHighestFleetRank;
filteredLabel.text = StringUtil.commaFormatNumber(data.avgHighestFleetRating);
break;
case LeaderboardEnum.WINS:
_rank = _data.winsRank;
filteredLabel.text = StringUtil.commaFormatNumber(data.wins);
break;
case LeaderboardEnum.KILL_DEATH_RATIO:
localization = Localization.instance;
_rank = _data.kdrRank;
filteredLabel.text = String(data.ktdRatio);
tooltip = localization.getStringWithTokens(_winLossTooltipText, {"[[Number.Kills]]":data.wins,
"[[Number.Losses]]":data.losses});
break;
case LeaderboardEnum.BLUEPRINT_PARTS:
_rank = _data.blueprintPartsRank;
filteredLabel.text = StringUtil.commaFormatNumber(data.blueprintPartsCollected);
break;
case LeaderboardEnum.QUALIFIED_WINS:
_rank = _data.qualifiedWinsPvPRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.qualifiedWinsPvP);
break;
case LeaderboardEnum.BUBBLE_HOUR_GRANTED:
_rank = _data.BubbleHoursGrantedRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.BubbleHoursGranted);
break;
case LeaderboardEnum.PVP_EVENT:
_rank = _data.currentPVPEventRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.currentPVPEvent);
break;
case LeaderboardEnum.PVP_EVENT_QUARTER:
_rank = _data.currentPVPEventQuarterRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.currentPVPEventQuarter);
break;
case
```

```
LeaderboardEnum.CREDITS_TRADE_ROUTE:
_rank = _data.CreditsTradeRouteRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.CreditsTradeRoute);
break;
case LeaderboardEnum.RESOURCE_TRADE_ROUTE:
_rank = _data.ResourcesTradeRouteRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.ResourcesTradeRoute);
break;
case LeaderboardEnum.RESOURCE_SALVAGED:
_rank = _data.ResourcesSalvagedRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.ResourcesSalvaged);
break;
case LeaderboardEnum.CREDITS_BOUNTY:
_rank = _data.CreditsBountyRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.CreditsBounty);
break;
case LeaderboardEnum.WINS_VS_BASES:
_rank = _data.WinsVsBaseRank;
filteredLabel.text = StringUtil.commaFormatNumber(_data.WinsVsBase);
break;
}

filteredLabel.width = filteredLabel.textWidth + 5;
filteredLabel.x = lastTypeXPos + ((width - lastTypeXPos) - filteredLabel.width) * 0.5 + offset;

if (tooltip != '')
_tooltip.addTooltip(_tooltipHitArea, this, null, tooltip);

_entryOne.text = String(rank);
_entryOne.textColor = (_isCurrentUser) ? 0xf7c78b : 0xf0f0f0
_entryOne.width = _entryOne.textWidth + 5;
_entryOne.x = (50 - _entryOne.width) * 0.5;
}

public function set index( v:uint ):void
{
if (_bg != null)
{
removeChild(_bg)
_bg = null;
}

if (v % 2 != 0 || _isCurrentUser)
{
_bg = UIFactory.getScaleBitmap((_isCurrentUser) ?
PanelEnum.LEADERBOARD_ROW_GLOW : PanelEnum.LEADERBOARD_ROW);
_bg.width = _width;
addChildAt(_bg, 0);
}
}

private
```

```
function onShowProfileClick( e:MouseEvent ):void
{
if (_data)
onClick.dispatch(_data.key);
}

public function get rank():int { return _rank; }

override public function get height():Number { return HEIGHT; }
override public function get width():Number { return _width; }

[Inject]
public function set tooltips( v:Tooltips ):void { _tooltip = v; }

public function destroy():void
{

_bg = null;

if (_entryOne)
_entryOne.destroy();
_entryOne = null;

if (_entryTwo)
_entryTwo.destroy();
_entryTwo = null;

if (_entryThree)
_entryThree.destroy();
_entryThree = null;

if (_entryFour)
_entryFour.destroy();
_entryFour = null;


if (_entryFive)
_entryFive.destroy();
_entryFive = null;

if (_image)
ObjectPool.give(_image);

_image = null;

if (_tooltip && _tooltipHitArea)
_tooltip.removeTooltip(_tooltipHitArea);

_tooltipHitArea = null;

_tooltip
```

```
= null;
}
}
}

------------------------------------
File 764: igw\com\ui\hud\shared\leaderboards\LeaderboardFilter.as
package com.ui.hud.shared.leaderboards
{
import com.ui.core.component.label.Label;

import flash.display.Sprite;

public class LeaderboardFilter extends Sprite
{
private var _text:Label;

public function LeaderboardFilter( fontSize:Number = 12, color:uint = 0, maxWidth:Number =
100, maxHeight:Number = 20, useLocalization:Boolean = true, fontNr:int = 0 )
{
_text = new Label(fontSize, color, maxWidth, maxHeight, useLocalization, fontNr);
addChild(_text);
}

public function set align( type:String ):void
{
_text.align = type;
}

public function set autoSize( autoSize:String ):void
{
_text.autoSize = autoSize;
}

public function set multiline( multiline:Boolean ):void
{
_text.multiline = multiline;
}

public function set text( value:String ):void
{
_text.text = value;
}

public function set htmlText( value:String ):void
{
_text.htmlText = value;
}

public function setTextWithTokens( value:String, tokens:Object ):void
{
```

```
_text.setTextWithTokens(value, tokens);
}

public function setHtmlTextWithTokens( value:String, tokens:Object ):void
{
_text.setHtmlTextWithTokens(value, tokens);
}

public function set textColor( v:uint ):void
{
_text.textColor = v;
}

public function set constrictTextToSize( v:Boolean ):void
{
_text.constrictTextToSize = v;
}

public function destroy():void
{
_text.destroy();
_text = null;
}
}
}

-------------------------------------
File 765: igw\com\ui\hud\shared\leaderboards\LeaderboardView.as
package com.ui.hud.shared.leaderboards
{
import com.enum.FactionEnum;
import com.enum.LeaderboardEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.leaderboards.LeaderboardEntryVO;
import com.model.leaderboards.LeaderboardVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.ILeaderboardPresenter;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.accordian.AccordianComponent;
import com.ui.core.component.accordian.AccordianGroup;
import
```

```
com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.ui.modal.playerinfo.PlayerProfileView;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import org.greensock.TweenLite;
import org.shared.ObjectPool;

public class LeaderboardView extends View
{
private var _bg:DefaultWindowBG;

private var _accordian:AccordianComponent;

private var _gridHolder:Sprite;
private var _container:Sprite;
private var _playerHolder:Sprite;

private var _generalGrid:Bitmap;
private var _allianceGrid:Bitmap;
private var _noMemberAllianceGrid:Bitmap;

private var _gridEnd:ScaleBitmap;

private var _subTitle:Label;
private var _noGridText:Label;
private var _titleOne:Label;
private var _titleTwo:Label;
private var _titleThree:Label;
private var _titleFour:Label;
private var _titleFive:Label;

private var _scrollbar:VScrollbar;

private var _scrollRect:Rectangle;

private var _maxHeight:int;

private var _groupID:String;
private var _subItemID:String;

private
```

```
var _players:Vector.<LeaderboardEntry>;

private var _titleText:String = 'CodeString.Leaderboard.Title'; //LEADERBOARDS
private var _subTitleText:String = 'CodeString.Leaderboard.SubTitle'; //UPDATES EVERY 24
HOURS
private var _noEntriesText:String = 'CodeString.Leaderboard.NoEntries'; //NO ENTRIES TO
DISPLAY
private var _noEntriesAllianceText:String = 'CodeString.Leaderboard.AllianceNoEntries'; //YOU
MUST HAVE ATLEAST 5 MEMBERS TO BE RANKED

private var _topPlayersText:String = 'CodeString.Leaderboard.Scope.TopPlayers'; //TOP
PLAYERS
private var _myRankingText:String = 'CodeString.Leaderboard.Scope.MyRanking'; //MY
RANKING
private var _mySectorText:String = 'CodeString.Leaderboard.Scope.MySector'; //MY SECTOR
private var _myAllianceText:String = 'CodeString.Leaderboard.Scope.MyAlliance'; //MY
ALLIANCE
private var _topAllianceText:String = 'CodeString.Leaderboard.Scope.TopAlliances'; //TOP
ALLIANCES
private var _myAllianceRankingText:String =
'CodeString.Leaderboard.Scope.MyAllianceRanking'; //MY ALLIANCE RANKING

private var _rankText:String = 'CodeString.Leaderboard.Type.Rank'; //Rank
private var _nameText:String = 'CodeString.Leaderboard.Type.Name'; //Name
private var _allianceText:String = 'CodeString.Leaderboard.Type.Alliance'; //Alliance
private var _sectorText:String = 'CodeString.Leaderboard.Type.Sector'; //Sector
private var _baseRatingText:String = 'CodeString.Leaderboard.Type.BaseRating'; //Base Rating
private var _levelText:String = 'CodeString.Leaderboard.Type.Level'; //Level
private var _commendationRankText:String =
'CodeString.Leaderboard.Type.CommendationRank'; //Commendation Rank
private var _totalCommendationRatingText:String =
'CodeString.Leaderboard.Type.TotalCommendationRating'; //Total Commendation Rating
private var _highestFleetRatingText:String =
'CodeString.Leaderboard.Type.HighestFleetRating'; //Highest Fleet Rating
private var _totalHighestFleetRatingText:String =
'CodeString.Leaderboard.Type.TotalHighestFleetRating' //Total Highest Fleet Rating
private var _victoriesText:String = 'CodeString.Leaderboard.Type.Victories'; //Victories
private var _winLossText:String = 'CodeString.Leaderboard.Type.WinLoss'; //Win/Loss Ratio
private var _totalBlueprintPartsText:String = 'CodeString.Leaderboard.Type.TotalBlueprintParts';
//Total Blueprint Parts
private var _numOfMembersText:String = 'CodeString.Leaderboard.Type.NumOfMembers';
//Number of Members
private var _totalExperienceText:String = 'CodeString.Leaderboard.Type.TotalExperience';
//Total Experience
private var _totalBaseRatingText:String = 'CodeString.Leaderboard.Type.TotalBaseRating';
//Total Base Rating
private var _avgHighestFleetText:String = 'CodeString.Leaderboard.Type.AvgHighestFleet';
//Average Highest Fleet Rating

private var _qualifiedWinsPvP:String = 'CodeString.Leaderboard.Type.QualifiedWinsPvP';
private
```

```
var _BubbleHoursGranted:String = 'CodeString.Leaderboard.Type.BubbleHoursGranted';
private var _currentPVPEvent:String = 'CodeString.Leaderboard.Type.CurrentPVPEvent';
private var _currentPVPEventQuarter:String =
'CodeString.Leaderboard.Type.CurrentPVPEventQuarter';
private var _CreditsTradeRoute:String = 'CodeString.Leaderboard.Type.CreditsTradeRoute';
private var _ResourcesTradeRoute:String =
'CodeString.Leaderboard.Type.ResourcesTradeRoute';
private var _ResourcesSalvaged:String = 'CodeString.Leaderboard.Type.ResourcesSalvaged';
private var _CreditsBounty:String = 'CodeString.Leaderboard.Type.CreditsBounty';
private var _WinsVsBase:String = 'CodeString.Leaderboard.Type.WinsVsBase';

[PostConstruct]
override public function init():void
{
super.init();

var playerFaction:IPrototype = presenter.getFactionPrototypesByName(CurrentUser.faction);
var playerFactionAssetVO:AssetVO = presenter.getAssetVOFromIPrototype(playerFaction);
var factionColor:uint = CommonFunctionUtil.getFactionColor(playerFaction.name);

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(1020, 593);
_bg.addTitle(_titleText, 470);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_subTitle = new Label(26, 0xffd785, 352, 45);
_subTitle.align = TextFormatAlign.RIGHT;
_subTitle.x = 140;
_subTitle.y = 6;
_subTitle.text = _subTitleText;

_noGridText = new Label(26, 0xf0f0f0, 352, 45);
_noGridText.align = TextFormatAlign.CENTER;
_noGridText.text = _noEntriesText;
_noGridText.x = 301 + (558 - _subTitle.textWidth) * 0.5;
_noGridText.y = 78 + (546 - _subTitle.height) * 0.5;
_noGridText.visible = false;

_accordian = ObjectPool.get(AccordianComponent);
_accordian.init(244, 42);
_accordian.x = _bg.bg.x + 14;
_accordian.y = _bg.bg.y - 10;
_accordian.addListener(onAccordianSelected);

_groupID = String(LeaderboardEnum.PLAYER_GLOBAL);
_accordian.addGroup(_groupID, _topPlayersText);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BASE_RATING),
_baseRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.EXPERIENCE),
_levelText, 0);
_accordian.addSubItemToGroup(_groupID,
```

```
String(LeaderboardEnum.COMMENDATION_COMBINED), _commendationRankText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.HIGHEST_FLEET_RATING), _highestFleetRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS), _victoriesText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.KILL_DEATH_RATIO),
_winLossText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BLUEPRINT_PARTS),
_totalBlueprintPartsText, 0);

_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.QUALIFIED_WINS),
_qualifiedWinsPvP, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.BUBBLE_HOUR_GRANTED), _BubbleHoursGranted, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.PVP_EVENT),
_currentPVPEvent, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.PVP_EVENT_QUARTER), _currentPVPEventQuarter, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.CREDITS_TRADE_ROUTE), _CreditsTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_TRADE_ROUTE), _ResourcesTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_SALVAGED), _ResourcesSalvaged, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.CREDITS_BOUNTY),
_CreditsBounty, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS_VS_BASES),
_WinsVsBase, 0);


_groupID = String(LeaderboardEnum.PLAYER_PERSONAL);
_accordian.addGroup(_groupID, _myRankingText);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BASE_RATING),
_baseRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.EXPERIENCE),
_levelText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.COMMENDATION_COMBINED), _commendationRankText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.HIGHEST_FLEET_RATING), _highestFleetRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS), _victoriesText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.KILL_DEATH_RATIO),
_winLossText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BLUEPRINT_PARTS),
_totalBlueprintPartsText, 0);

_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.QUALIFIED_WINS),
_qualifiedWinsPvP, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.BUBBLE_HOUR_GRANTED), _BubbleHoursGranted, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.PVP_EVENT),
_currentPVPEvent,
```

```
0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.PVP_EVENT_QUARTER), _currentPVPEventQuarter, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.CREDITS_TRADE_ROUTE), _CreditsTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_TRADE_ROUTE), _ResourcesTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_SALVAGED), _ResourcesSalvaged, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.CREDITS_BOUNTY),
_CreditsBounty, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS_VS_BASES),
_WinsVsBase, 0);


_groupID = String(LeaderboardEnum.PLAYER_SECTOR);
_accordian.addGroup(_groupID, _mySectorText);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BASE_RATING),
_baseRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.EXPERIENCE),
_levelText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.COMMENDATION_COMBINED), _commendationRankText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.HIGHEST_FLEET_RATING), _highestFleetRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS), _victoriesText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.KILL_DEATH_RATIO),
_winLossText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BLUEPRINT_PARTS),
_totalBlueprintPartsText, 0);

_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.QUALIFIED_WINS),
_qualifiedWinsPvP, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.BUBBLE_HOUR_GRANTED), _BubbleHoursGranted, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.PVP_EVENT),
_currentPVPEvent, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.PVP_EVENT_QUARTER), _currentPVPEventQuarter, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.CREDITS_TRADE_ROUTE), _CreditsTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_TRADE_ROUTE), _ResourcesTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_SALVAGED), _ResourcesSalvaged, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.CREDITS_BOUNTY),
_CreditsBounty, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS_VS_BASES),
_WinsVsBase, 0);

if
```

```
(CurrentUser.alliance != '')
{
_groupID = String(LeaderboardEnum.PLAYER_ALLIANCE);
_accordian.addGroup(_groupID, _myAllianceText);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BASE_RATING),
_baseRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.EXPERIENCE),
_levelText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.COMMENDATION_COMBINED), _commendationRankText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.HIGHEST_FLEET_RATING), _highestFleetRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS), _victoriesText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.KILL_DEATH_RATIO),
_winLossText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BLUEPRINT_PARTS),
_totalBlueprintPartsText, 0);

_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.QUALIFIED_WINS),
_qualifiedWinsPvP, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.BUBBLE_HOUR_GRANTED), _BubbleHoursGranted, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.PVP_EVENT),
_currentPVPEvent, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.PVP_EVENT_QUARTER), _currentPVPEventQuarter, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.CREDITS_TRADE_ROUTE), _CreditsTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_TRADE_ROUTE), _ResourcesTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_SALVAGED), _ResourcesSalvaged, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.CREDITS_BOUNTY),
_CreditsBounty, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS_VS_BASES),
_WinsVsBase, 0);

}

_groupID = String(LeaderboardEnum.ALLIANCE_GLOBAL);
_accordian.addGroup(_groupID, _topAllianceText);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BASE_RATING),
_baseRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.EXPERIENCE),
_totalExperienceText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.COMMENDATION_COMBINED), _totalCommendationRatingText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.HIGHEST_FLEET_RATING), _totalHighestFleetRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS), _victoriesText, 0);
_accordian.addSubItemToGroup(_groupID,
```

```
String(LeaderboardEnum.KILL_DEATH_RATIO), _winLossText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BLUEPRINT_PARTS),
_totalBlueprintPartsText, 0);

_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.ALLIANCE_NUM_MEMBERS), _numOfMembersText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.ALLIANCE_TOTAL_RATING), _totalBaseRatingText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.ALLIANCE_AVERAGE_HIGHEST_FLEET_RATING),
_avgHighestFleetText, 0);

_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.QUALIFIED_WINS),
_qualifiedWinsPvP, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.BUBBLE_HOUR_GRANTED), _BubbleHoursGranted, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.PVP_EVENT),
_currentPVPEvent, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.PVP_EVENT_QUARTER), _currentPVPEventQuarter, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.CREDITS_TRADE_ROUTE), _CreditsTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_TRADE_ROUTE), _ResourcesTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_SALVAGED), _ResourcesSalvaged, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.CREDITS_BOUNTY),
_CreditsBounty, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS_VS_BASES),
_WinsVsBase, 0);


if (CurrentUser.alliance != '')
{
_groupID = String(LeaderboardEnum.ALLIANCE_PERSONAL);
_accordian.addGroup(_groupID, _myAllianceRankingText);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BASE_RATING),
_baseRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.EXPERIENCE),
_totalExperienceText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.COMMENDATION_COMBINED), _totalCommendationRatingText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.HIGHEST_FLEET_RATING), _totalHighestFleetRatingText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS), _victoriesText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.KILL_DEATH_RATIO),
_winLossText, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.BLUEPRINT_PARTS),
_totalBlueprintPartsText, 0);

_accordian.addSubItemToGroup(_groupID,
```

```
String(LeaderboardEnum.ALLIANCE_NUM_MEMBERS), _numOfMembersText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.ALLIANCE_TOTAL_RATING), _totalBaseRatingText, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.ALLIANCE_AVERAGE_HIGHEST_FLEET_RATING),
_avgHighestFleetText, 0);

_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.QUALIFIED_WINS),
_qualifiedWinsPvP, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.BUBBLE_HOUR_GRANTED), _BubbleHoursGranted, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.PVP_EVENT),
_currentPVPEvent, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.PVP_EVENT_QUARTER), _currentPVPEventQuarter, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.CREDITS_TRADE_ROUTE), _CreditsTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_TRADE_ROUTE), _ResourcesTradeRoute, 0);
_accordian.addSubItemToGroup(_groupID,
String(LeaderboardEnum.RESOURCE_SALVAGED), _ResourcesSalvaged, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.CREDITS_BOUNTY),
_CreditsBounty, 0);
_accordian.addSubItemToGroup(_groupID, String(LeaderboardEnum.WINS_VS_BASES),
_WinsVsBase, 0);

}

_accordian.setSelected(String(LeaderboardEnum.PLAYER_GLOBAL),
String(LeaderboardEnum.BASE_RATING));

_container = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED,
PanelEnum.HEADER_NOTCHED_RIGHT, 750, 548, 30, _accordian.x + _accordian.width + 5,
_accordian.y, "BASE RATING", LabelEnum.
H3);
Label(_container.getChildAt(2)).allCaps = true;

_generalGrid = UIFactory.getBitmap("LeaderboardGeneralGridBMD");
_allianceGrid = UIFactory.getBitmap("LeaderboardAllianceGridBMD");
_noMemberAllianceGrid = UIFactory.getBitmap("LeaderboardAllianceMemberGridBMD");

_titleOne = new Label(16, 0xf0f0f0, 51, 30);
_titleOne.align = TextFormatAlign.CENTER;
_titleOne.allCaps = true;
_titleOne.text = _rankText;
_titleOne.y = -_titleOne.height;

_titleTwo = new Label(16, 0xf0f0f0, 215, 30);
_titleTwo.align = TextFormatAlign.CENTER;
_titleTwo.allCaps = true;
_titleTwo.text
```

```
= _nameText;
_titleTwo.y = -_titleOne.height;

_titleThree = new Label(16, 0xf0f0f0, 215, 30);
_titleThree.align = TextFormatAlign.CENTER;
_titleThree.allCaps = true;

_titleFour = new Label(16, 0xf0f0f0, 215, 30);
_titleFour.align = TextFormatAlign.CENTER;
_titleFour.allCaps = true;

_titleFive = new Label(16, 0xf0f0f0, 215, 30);
_titleFive.align = TextFormatAlign.CENTER;
_titleFour.allCaps = true;

_gridEnd = UIFactory.getScaleBitmap(PanelEnum.GRID_END);
_players = new Vector.<LeaderboardEntry>;

_playerHolder = new Sprite();
_gridHolder = new Sprite();

_maxHeight = 0;

_scrollRect = new Rectangle(0, 0, 632, 519);

var dragBarBGRect:Rectangle = new Rectangle(0, 3, 5, 7);
_scrollbar = new VScrollbar();
_scrollbar.init(7, 480, 0, 0, dragBarBGRect, '', 'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 40;

addChild(_bg);
addChild(_subTitle);
addChild(_accordian);
addChild(_container);
addChild(_gridHolder);
addChild(_playerHolder);
addChild(_scrollbar);
addChild(_titleOne);
addChild(_titleTwo);
addChild(_titleThree);
addChild(_titleFour);
addChild(_titleFive);
addChild(_noGridText);

presenter.addOnLeaderboardDataUpdatedListener(leaderboardUpdate);

addEffects();
effectsIN();
```

```
onAccordianSelected(String(LeaderboardEnum.PLAYER_GLOBAL),
String(LeaderboardEnum.BASE_RATING), null);
}

private function onAccordianSelected( groupID:String, subItemID:String, data:* ):void
{
_playerHolder.alpha = 0;
_gridHolder.alpha = 0;
_scrollbar.alpha = 0;

_titleOne.alpha = 0;
_titleTwo.alpha = 0;
_titleThree.alpha = 0;
_titleFour.alpha = 0;
_titleFive.alpha = 0;

var group:AccordianGroup = _accordian.getGroup(groupID);
if (!subItemID && group.hasSubItems)
{
subItemID = group.subItems[0].id;
_accordian.setSelected(groupID, subItemID)
}

_groupID = groupID;
_subItemID = subItemID;

presenter.currentLeaderboardScope = int(_groupID);
presenter.currentLeaderboardType = int(_subItemID);

setUpGrid();

var leaderboardData:LeaderboardVO = presenter.getLeaderboardData();

if (leaderboardData)
leaderboardUpdate(leaderboardData);
}

private function setUpGrid():void
{
var bitmap:Bitmap;
var endBitmap:Bitmap;
var finalTitle:Label;

_playerHolder.visible = true;
_gridHolder.visible = true;
_scrollbar.visible = true;

_titleOne.visible = true;
_titleTwo.visible
```

```
= true;
_titleThree.visible = true;
_titleFour.visible = true;
_titleFive.visible = true;

_noGridText.visible = false;

_gridHolder.removeChildren();

switch (int(_groupID))
{
case LeaderboardEnum.PLAYER_GLOBAL:
case LeaderboardEnum.PLAYER_PERSONAL:
case LeaderboardEnum.PLAYER_SECTOR:
case LeaderboardEnum.PLAYER_ALLIANCE:
bitmap = _generalGrid;

_titleTwo.width = 120;

_titleThree.text = _allianceText;
_titleThree.x = 170;
_titleThree.width = 225;

_titleFour.text = _sectorText;
_titleFour.x = 394;
_titleFour.width = 126;

_titleFive.visible = true;
_titleFour.visible = true;

finalTitle = _titleFive;
break;
case LeaderboardEnum.ALLIANCE_GLOBAL:
case LeaderboardEnum.ALLIANCE_PERSONAL:
_titleFive.visible = false;
_titleTwo.width = 225;
if (int(_subItemID) == LeaderboardEnum.ALLIANCE_NUM_MEMBERS)
{
bitmap = _noMemberAllianceGrid;
_titleFour.visible = false;
finalTitle = _titleThree;
} else
{
bitmap = _allianceGrid;

_titleThree.text = _numOfMembersText;
_titleThree.x = 275;
_titleThree.width = 150;

_titleFour.visible = true;

finalTitle
```

```
= _titleFour;

}
break;
}

_gridHolder.addChildAt(bitmap, 0);
var stringToUse:String = '';
switch (int(_subItemID))
{
case LeaderboardEnum.BASE_RATING:
setContainerTitle(_baseRatingText);
finalTitle.text = _baseRatingText;
break;
case LeaderboardEnum.EXPERIENCE:
if (int(_groupID) == LeaderboardEnum.ALLIANCE_GLOBAL || int(_groupID) ==
LeaderboardEnum.ALLIANCE_PERSONAL)
stringToUse = _totalExperienceText;
else
stringToUse = _levelText;

setContainerTitle(stringToUse);
finalTitle.text = stringToUse;
break;
case LeaderboardEnum.COMMENDATION_COMBINED:
if (int(_groupID) == LeaderboardEnum.ALLIANCE_GLOBAL || int(_groupID) ==
LeaderboardEnum.ALLIANCE_PERSONAL)
stringToUse = _totalCommendationRatingText;
else
stringToUse = _commendationRankText;

setContainerTitle(stringToUse);
finalTitle.text = stringToUse;
break;
case LeaderboardEnum.HIGHEST_FLEET_RATING:
if (int(_groupID) == LeaderboardEnum.ALLIANCE_GLOBAL || int(_groupID) ==
LeaderboardEnum.ALLIANCE_PERSONAL)
stringToUse = _totalHighestFleetRatingText;
else
stringToUse = _highestFleetRatingText;

setContainerTitle(stringToUse);
finalTitle.text = stringToUse;
break;
case LeaderboardEnum.WINS:
setContainerTitle(_victoriesText);
finalTitle.text = _victoriesText;
break;
case LeaderboardEnum.KILL_DEATH_RATIO:
setContainerTitle(_winLossText);
finalTitle.text
```

```
= _winLossText;
break;
case LeaderboardEnum.BLUEPRINT_PARTS:
setContainerTitle(_totalBlueprintPartsText);
finalTitle.text = _totalBlueprintPartsText;
break;
case LeaderboardEnum.ALLIANCE_NUM_MEMBERS:
setContainerTitle(_numOfMembersText);
finalTitle.text = _numOfMembersText;
break;
case LeaderboardEnum.ALLIANCE_TOTAL_RATING:
setContainerTitle(_totalBaseRatingText);
finalTitle.text = _totalBaseRatingText;
break;
case LeaderboardEnum.ALLIANCE_AVERAGE_HIGHEST_FLEET_RATING:
setContainerTitle(_avgHighestFleetText);
finalTitle.text = _avgHighestFleetText;
break;
case LeaderboardEnum.QUALIFIED_WINS:
setContainerTitle(_qualifiedWinsPvP);
finalTitle.text = _qualifiedWinsPvP;
break;
case LeaderboardEnum.BUBBLE_HOUR_GRANTED:
setContainerTitle(_BubbleHoursGranted);
finalTitle.text = _BubbleHoursGranted;
break;
case LeaderboardEnum.PVP_EVENT:
setContainerTitle(_currentPVPEvent);
finalTitle.text = _currentPVPEvent;
break;
case LeaderboardEnum.PVP_EVENT_QUARTER:
setContainerTitle(_currentPVPEventQuarter);
finalTitle.text = _currentPVPEventQuarter;
break;
case LeaderboardEnum.CREDITS_TRADE_ROUTE:
setContainerTitle(_CreditsTradeRoute);
finalTitle.text = _CreditsTradeRoute;
break;
case LeaderboardEnum.RESOURCE_TRADE_ROUTE:
setContainerTitle(_ResourcesTradeRoute);
finalTitle.text = _ResourcesTradeRoute;
break;
case LeaderboardEnum.RESOURCE_SALVAGED:
setContainerTitle(_ResourcesSalvaged);
finalTitle.text = _ResourcesSalvaged;
break;
case LeaderboardEnum.CREDITS_BOUNTY:
setContainerTitle(_CreditsBounty);
finalTitle.text = _CreditsBounty;
break;
case
```

```
LeaderboardEnum.WINS_VS_BASES:
setContainerTitle(_WinsVsBase);
finalTitle.text = _WinsVsBase;
break;
}

_titleOne.x = 0;
_titleTwo.x = 51;

var gridEndWidth:int = 717 - _gridHolder.width;

_gridEnd.width = gridEndWidth;
_gridEnd.x = bitmap.x + bitmap.width;
_gridHolder.addChild(_gridEnd);

finalTitle.x = _gridEnd.x;
finalTitle.y = -finalTitle.textHeight;
finalTitle.width = _gridEnd.width;

_playerHolder.x = _gridHolder.x = 360 + (558 - _gridHolder.width) * 0.5;
_playerHolder.y = _gridHolder.y = 78 + (546 - _gridHolder.height) * 0.5;

_titleOne.x += _gridHolder.x;
_titleTwo.x += _gridHolder.x;
_titleThree.x += _gridHolder.x;
_titleFour.x += _gridHolder.x;
_titleFive.x += _gridHolder.x;

_titleOne.y = _titleTwo.y = _titleThree.y = _titleFour.y = _titleFive.y = _gridHolder.y - 20;

_scrollbar.x = _gridHolder.x + _gridHolder.width + 4;
_scrollbar.y = _gridHolder.y;
_scrollbar.resetScroll();

_scrollRect = new Rectangle(0, 0, _gridHolder.width, _gridHolder.height - 1);
_scrollRect.y = 0;
_playerHolder.scrollRect = _scrollRect;


if (_gridHolder.alpha == 0)
{
TweenLite.to(_playerHolder, 0.5, {alpha:1});
TweenLite.to(_gridHolder, 0.5, {alpha:1});
TweenLite.to(_scrollbar, 0.5, {alpha:1});

TweenLite.to(_titleOne, 0.5, {alpha:1});
TweenLite.to(_titleTwo, 0.5, {alpha:1});
TweenLite.to(_titleThree, 0.5, {alpha:1});
TweenLite.to(_titleFour, 0.5, {alpha:1});
TweenLite.to(_titleFive, 0.5, {alpha:1});
}
```

```
}

public function leaderboardUpdate( leaderboardData:LeaderboardVO ):void
{
var entries:Vector.<LeaderboardEntryVO>;
if (int(_groupID) == LeaderboardEnum.ALLIANCE_GLOBAL || int(_groupID) ==
LeaderboardEnum.ALLIANCE_PERSONAL)
entries = leaderboardData.alliances;
else
entries = leaderboardData.players;

var len:uint = entries.length;
if (len > 0)
{
var currentUserKey:String = CurrentUser.id;
var currentPlayerEntry:LeaderboardEntry;
var currentPlayer:LeaderboardEntryVO;
var currentRace:IPrototype;
var faction:String;
var assetName:String;
var factionColor:uint;
var rank:int;
var rankProto:IPrototype;
var rankAssetVO:AssetVO;
var i:uint;

while (_players.length > len)
{
currentPlayerEntry = _players.shift();
if (_playerHolder.contains(currentPlayerEntry))
_playerHolder.removeChild(currentPlayerEntry);

currentPlayerEntry.destroy();
currentPlayerEntry = null;
}

var currentPlayersLen:uint = _players.length;
var currentUser:LeaderboardEntry;
var width:int = _gridHolder.width;
var lastTypeXPos:int = _gridHolder.getChildAt(0).width;
for (; i < len; ++i)
{
currentPlayer = entries[i];
var isCurrentUser:Boolean = (currentPlayer.key == currentUserKey);
if (currentPlayer.isAlliance)
faction = currentPlayer.racePrototype;
else
currentRace = presenter.getRacePrototypeByName(currentPlayer.racePrototype);

if
```

```
(currentRace)
faction = currentRace.getUnsafeValue('faction');

factionColor = CommonFunctionUtil.getFactionColor(faction);

switch (faction)
{
case FactionEnum.IGA:
assetName = 'igaUIAsset';
break;
case FactionEnum.SOVEREIGNTY:
assetName = 'sovUIAsset';
break;
case FactionEnum.TYRANNAR:
assetName = 'tryUIAsset';
break;
}

rank = CommonFunctionUtil.getCommendationRank(currentPlayer.commendationPointsPvE +
currentPlayer.commendationPointsPvP);
rankProto =
presenter.getCommendationRankPrototypesByName(CommonFunctionUtil.getCommendationProtoName
rankAssetVO = presenter.getAssetVO(rankProto.getValue(assetName));


if (i < currentPlayersLen)
{
currentPlayerEntry = _players[i];
if (currentPlayerEntry != null)
currentPlayerEntry.update(width, lastTypeXPos, int(_groupID), int(_subItemID), currentPlayer,
factionColor, rankAssetVO, isCurrentUser);
else
{
currentPlayerEntry = new LeaderboardEntry(presenter);
currentPlayerEntry.onClick.add(onShowProfile);
presenter.injectObject(currentPlayerEntry);
currentPlayerEntry.update(width, lastTypeXPos, int(_groupID), int(_subItemID), currentPlayer,
factionColor, rankAssetVO, isCurrentUser);
_players.push(currentPlayerEntry);
}
} else
{
currentPlayerEntry = new LeaderboardEntry(presenter);
currentPlayerEntry.onClick.add(onShowProfile);
presenter.injectObject(currentPlayerEntry);
currentPlayerEntry.update(width, lastTypeXPos, int(_groupID), int(_subItemID), currentPlayer,
factionColor, rankAssetVO, isCurrentUser);
_players.push(currentPlayerEntry);
}


if
```

```
(!isCurrentUser)
_playerHolder.addChild(currentPlayerEntry);
else
currentUser = currentPlayerEntry;
}
if (currentUser)
_playerHolder.addChildAt(currentUser, _playerHolder.numChildren);

layout();
} else
{
switch (int(_groupID))
{
case LeaderboardEnum.ALLIANCE_PERSONAL:
_noGridText.text = _noEntriesAllianceText;
break;
default:
_noGridText.text = _noEntriesText;
break;
}
_noGridText.x = 301 + (558 - _subTitle.textWidth) * 0.5;

_playerHolder.visible = false;
_gridHolder.visible = false;
_scrollbar.visible = false;
_titleOne.visible = false;
_titleTwo.visible = false;
_titleThree.visible = false;
_titleFour.visible = false;
_titleFive.visible = false;
_noGridText.visible = true;
}
}

private function layout():void
{
_players.sort(orderEntriesByRank);
var len:uint = _players.length;
var selection:LeaderboardEntry;
var yPos:int = 0;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _players[i];
selection.index = i + 1;
selection.y = yPos;
_maxHeight += selection.height - 1;
yPos += selection.height - 1;
}
_scrollbar.updateScrollableHeight(_maxHeight);
}
```

```
private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_playerHolder.scrollRect = _scrollRect;
}

private function setContainerTitle( v:String ):void { Label(_container.getChildAt(2)).text = v; }

private function orderEntriesByRank( entryOne:LeaderboardEntry, entryTwo:LeaderboardEntry
):Number
{
if (!entryOne)
return -1;
if (!entryTwo)
return 1;

var playerOneRank:int = entryOne.rank;
var playerTwoRank:int = entryTwo.rank;

if (playerOneRank > playerTwoRank)
return 1;
else if (playerOneRank < playerTwoRank)
return -1;

return 0;
}

override public function get height():Number
{
return _bg.height;
}

override public function get width():Number
{
return _bg.width;
}

private function onShowProfile( v:String ):void
{
var playerProfileView:PlayerProfileView =
PlayerProfileView(_viewFactory.createView(PlayerProfileView));
playerProfileView.playerKey = v;
_viewFactory.notify(playerProfileView);
}

[Inject]
public function set presenter( v:ILeaderboardPresenter ):void { _presenter = v; }
public function get presenter():ILeaderboardPresenter { return
ILeaderboardPresenter(_presenter);
```

```
}

override public function destroy():void
{
presenter.removeOnLeaderboardDataUpdatedListener(leaderboardUpdate);
super.destroy();

var len:uint = _players.length;
var selection:LeaderboardEntry;
for (var i:uint = 0; i < len; ++i)
{
selection = _players[i];
selection.destroy();
}

_players.length = 0;

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_accordian)
ObjectPool.give(_accordian);

_accordian = null;

if (_playerHolder)
TweenLite.killTweensOf(_playerHolder);

_playerHolder = null;

if (_gridHolder)
TweenLite.killTweensOf(_gridHolder);

_gridHolder = null;

if (_scrollbar)
{
TweenLite.killTweensOf(_scrollbar);
_scrollbar.destroy();
}

_scrollbar = null;

if (_subTitle)
_subTitle.destroy();

_subTitle = null;

if
```

```
(_noGridText)
_noGridText.destroy();

_noGridText = null;

if (_titleOne)
{
TweenLite.killTweensOf(_titleOne);
_titleOne.destroy();
}

_titleOne = null;

if (_titleTwo)
{
TweenLite.killTweensOf(_titleTwo);
_titleTwo.destroy();
}

_titleTwo = null;

if (_titleThree)
{
TweenLite.killTweensOf(_titleThree);
_titleThree.destroy();
}

_titleThree = null;

if (_titleFour)
{
TweenLite.killTweensOf(_titleFour);
_titleFour.destroy();
}

_titleFour = null;

if (_titleFive)
{
TweenLite.killTweensOf(_titleFive);
_titleFive.destroy();
}

_titleFive = null;
}
}
}
```

File 766: igw\com\ui\hud\shared\mail\MailBoxView.as

```
package
```

```
com.ui.hud.shared.mail
{
import com.model.mail.MailVO;
import com.presenter.shared.IUIPresenter;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

public class MailBoxView extends View
{
private var _bg:Bitmap;
private var _closeBtn:BitmapButton;
private var _messageFrame:Bitmap;
private var _mailEntries:Dictionary;
private var _title:Label;
private var _noMail:Label;

private var _deleteBtn:BitmapButton;
private var _selectionCheckbox:BitmapButton;

private var _checkboxEmpty:BitmapData;
private var _checkboxDash:BitmapData;
private var _checkboxChecked:BitmapData;

private var _selectionCount:int;

private var _mail:Vector.<MailEntry>;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _holder:Sprite;

private var _mailbox:String = 'CodeString.Mail.Mailbox'; //MAILBOX
private var _emptyMailbox:String = 'CodeString.Mail.NoMail'; //You have no mail - get some
friends. Or enemies.

[PostConstruct]
```

```
override public function init():void
{
super.init();

_mailEntries = new Dictionary;
_mail = new Vector.<MailEntry>;
var windowBGClass:Class = Class(getDefinitionByName('MailWindowMessageBMD'));
var messageFrameClass:Class = Class(getDefinitionByName('MailWindowMainBMD'));

var checkboxEmptyClass:Class = Class(getDefinitionByName('CheckboxBtnUncheckedBMD'));
var checkboxFilledClass:Class = Class(getDefinitionByName('CheckboxBtnCheckedBMD'));
var checkboxDashClass:Class = Class(getDefinitionByName('CheckboxBtnDeselectBMD'));

_bg = new Bitmap(BitmapData(new windowBGClass()));

_checkboxEmpty = BitmapData(new checkboxEmptyClass());
_checkboxDash = BitmapData(new checkboxDashClass());
_checkboxChecked = BitmapData(new checkboxFilledClass());

_title = new Label(20, 0xf0f0f0, 150, 25, true);
_title.align = TextFormatAlign.LEFT;
_title.x = 25;
_title.y = 12;
_title.text = _mailbox;

_messageFrame = new Bitmap(BitmapData(new messageFrameClass()));
_messageFrame.x = 18;
_messageFrame.y = 46;

_noMail = new Label(24, 0xf0f0f0);
_noMail.constrictTextToSize = false;
_noMail.autoSize = TextFieldAutoSize.CENTER;
_noMail.x = _bg.x + (_bg.width - _noMail.textWidth) * 0.5;
_noMail.y = _bg.y + (_bg.height - _noMail.textHeight) * 0.5;
_noMail.text = _emptyMailbox;

_deleteBtn = ButtonFactory.getBitmapButton('MailTrashBtnUpBMD', 598, 50, '', 0,
'MailTrashBtnRollOverBMD', 'MailTrashBtnSelectedBMD', null, 'MailTrashBtnSelectedBMD');
addListener(_deleteBtn, MouseEvent.CLICK, onDeleteMail);

_selectionCheckbox = ButtonFactory.getBitmapButton('CheckboxBtnUncheckedBMD', 32, 56, '',
0, 'CheckboxBtnUncheckedBMD');
addListener(_selectionCheckbox, MouseEvent.CLICK, onSelectionChanged);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

_holder = new Sprite();
_holder.x = 24;
_holder.y
```

```
= 90;
_maxHeight = 0;

_scrollRect = new Rectangle(0, 0, 666, 336);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect;

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 5, 5, 2);
var scrollbarXPos:Number = 692;
var scrollbarYPos:Number = 88;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 12;

addChild(_bg);
addChild(_closeBtn);
addChild(_title);
addChild(_messageFrame);
addChild(_noMail);
addChild(_scrollbar);
addChild(_deleteBtn);
addChild(_selectionCheckbox);
addChild(_holder);

presenter.addOnMailHeadersUpdatedListener(onMailHeadersUpdated);
presenter.addMailCountUpdateListener(onMailCountUpdated);
presenter.sendGetMailboxMessage();

addEffects();
effectsIN();
}

private function onMailHeadersUpdated( v:Vector.<MailVO> ):void
{
var len:uint = v.length;
var currentMailVO:MailVO;
var currentMailEntry:MailEntry;
for (var i:uint = 0; i < len; ++i)
{
currentMailVO = v[i];
if (currentMailVO.key in _mailEntries)
currentMailEntry = _mailEntries[currentMailVO.key];
else
{
currentMailEntry = new MailEntry();
currentMailEntry.init(currentMailVO);
currentMailEntry.onClicked.add(onEntryClicked);
```

```
currentMailEntry.onSelectionChanged.add(onMailSelectionChange);
_mail.push(currentMailEntry);
}
_mailEntries[currentMailVO.key] = currentMailEntry;
_holder.addChild(currentMailEntry);
}

_mail.sort(orderItems);
updateSelectionBtn();
layout();
}

private function onEntryClicked( entry:MailEntry ):void
{
presenter.mailRead(entry.mailKey);
var mailView:MailView = MailView(_viewFactory.createView(MailView));
_viewFactory.notify(mailView);
mailView.setMail(entry.mail, onDeletedFromMailView);

}

private function onDeleteMail( e:MouseEvent ):void
{
var len:uint = _mail.length;
var currentMailEntry:MailEntry;
var keysToRemove:Vector.<String> = new Vector.<String>;
for (var i:uint = 0; i < len; ++i)
{
currentMailEntry = _mail[i];
if (currentMailEntry.selected)
{
if (currentMailEntry.selected)
--_selectionCount;

_holder.removeChild(currentMailEntry);
delete _mailEntries[currentMailEntry.mailKey]
keysToRemove.push(currentMailEntry.mailKey);
_mail.splice(i, 1);
currentMailEntry = null;
--len;
--i;
}
}
presenter.deleteMail(keysToRemove);
updateSelectionBtn();
layout();
}

private function onDeletedFromMailView( key:String ):void
{
```

```
var len:uint = _mail.length;
var currentMailEntry:MailEntry;
var keysToRemove:Vector.<String> = new Vector.<String>;
for (var i:uint = 0; i < len; ++i)
{
currentMailEntry = _mail[i];
if (currentMailEntry.mailKey == key)
{
if (currentMailEntry.selected)
--_selectionCount;

_holder.removeChild(currentMailEntry);
delete _mailEntries[currentMailEntry.mailKey]
keysToRemove.push(currentMailEntry.mailKey);
_mail.splice(i, 1);
currentMailEntry = null;
break;
}
}
presenter.deleteMail(keysToRemove);
updateSelectionBtn();
layout();
}

protected function layout():void
{
var len:uint = _mail.length;
var selection:MailEntry;
var yPos:int = 0;
_noMail.visible = (len > 0) ? false : true;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _mail[i];
selection.y = yPos;
_maxHeight += selection.height + 4;
yPos += selection.height + 4;
}
_maxHeight -= 3;
_scrollbar.updateScrollableHeight(_maxHeight);
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

private function orderItems( itemOne:MailEntry, itemTwo:MailEntry ):Number
{
```

```
if (!itemOne)
return -1;
if (!itemTwo)
return 1;

var timeSentOne:Number = itemOne.timeSent;
var timeSent:Number = itemTwo.timeSent;

if (timeSentOne > timeSent)
return -1;
else if (timeSentOne < timeSent)
return 1;

return 0;
}

private function onMailSelectionChange( v:Boolean ):void
{
if (v)
++_selectionCount;
else
--_selectionCount;

updateSelectionBtn();
}

private function updateSelectionBtn():void
{
if (_selectionCount == _mail.length)
_selectionCheckbox.updateBackgrounds(_checkboxChecked, _checkboxChecked);
else if (_selectionCount > 0)
_selectionCheckbox.updateBackgrounds(_checkboxDash, _checkboxDash);
else
_selectionCheckbox.updateBackgrounds(_checkboxEmpty, _checkboxEmpty);
}

private function onSelectionChanged( e:MouseEvent ):void
{

if (_selectionCount == _mail.length)
setMailEntrySelected(false)
else if (_selectionCount > 0)
setMailEntrySelected(false)
else
setMailEntrySelected(true)
}

private function setMailEntrySelected( v:Boolean ):void
{
```

```
var len:uint = _mail.length;
var currentMailEntry:MailEntry;

if (v)
_selectionCount = len;
else
_selectionCount = 0;

for (var i:uint = 0; i < len; ++i)
{
currentMailEntry = _mail[i];
currentMailEntry.selected = v;
}
updateSelectionBtn();
}

private function onMailCountUpdated( unread:uint, count:uint, serverUpdate:Boolean ):void
{
if (serverUpdate)
presenter.sendGetMailboxMessage();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
presenter.removeOnMailHeadersUpdatedListener(onMailHeadersUpdated);
presenter.removeMailCountUpdateListener(onMailCountUpdated);
super.destroy();

_bg = null;
_closeBtn.destroy();
_closeBtn = null;
_messageFrame = null;

_title.destroy();
_title = null;

_noMail.destroy();
_noMail = null;

_deleteBtn.destroy();
_deleteBtn = null;

var
```

```
len:uint = _mail.length;
var currentMailEntry:MailEntry;
for (var i:uint = 0; i < len; ++i)
{
currentMailEntry = _mail[i];
_holder.removeChild(currentMailEntry);
currentMailEntry.destroy();
currentMailEntry = null;
}
_mail.length = 0;

for (var key:String in _mailEntries)
{
delete _mailEntries[key];
}
_mailEntries = null;

_holder = null;

_scrollbar.destroy();
_scrollbar = null;


_maxHeight = 0;
}
}
}
```

File 767: igw\com\ui\hud\shared\mail\MailEntry.as

```
package com.ui.hud.shared.mail
{
import com.model.mail.MailVO;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.ColorTransform;
import flash.text.TextFormatAlign;
import flash.ui.Mouse;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;

public class MailEntry extends Sprite
{
private
```

```
var _bg:Bitmap;
private var _clickBG:Sprite;
private var _hoverImage:Bitmap;

private var _sender:Label;
private var _subject:Label;
private var _dateSent:Label;

private var _isRead:Boolean;

private var _checkbox:BitmapButton;

private var _mail:MailVO;

public var onClicked:Signal;
public var onSelectionChanged:Signal;

public function MailEntry()
{
super();

onClicked = new Signal(MailEntry);
onSelectionChanged = new Signal(Boolean);

var windowBGClass:Class = Class(getDefinitionByName('MailMessageBlueRowBMD'));
var hoverImageClass:Class = Class(getDefinitionByName('MailTabRollOverBMD'));
_bg = new Bitmap(BitmapData(new windowBGClass()));

_clickBG = new Sprite();
_clickBG.addChild(_bg);
_clickBG.buttonMode = true;
_clickBG.useHandCursor = true;
_clickBG.x = 36;

_hoverImage = new Bitmap(BitmapData(new hoverImageClass()));
_hoverImage.visible = false;
_hoverImage.y = 15;

_checkbox = ButtonFactory.getBitmapButton('CheckboxBtnUncheckedBMD', 9, 12, '', 0, null,
'CheckboxBtnUncheckedBMD', null, 'CheckboxBtnCheckedBMD');
_checkbox.selectable = true;

_sender = new Label(14, 0xf0f0f0, 300, 25, false, 1);
_sender.x = 49;
_sender.y = 1;
_sender.align = TextFormatAlign.LEFT;

_subject = new Label(12, 0xf0f0f0, 600, 25, false, 1);
_subject.x = 49;
_subject.y = 22;
_subject.align
```

```
= TextFormatAlign.LEFT;

_dateSent = new Label(13, 0xf0f0f0, 300, 25, false, 1);
_dateSent.x = 349;
_dateSent.y = 1;
_dateSent.align = TextFormatAlign.RIGHT;

_clickBG.addEventListener(MouseEvent.CLICK, onClick, false, 0, true);
_clickBG.addEventListener(MouseEvent.ROLL_OUT, onMouseRollOut, false, 0, true);
_clickBG.addEventListener(MouseEvent.ROLL_OVER, onMouseRollOver, false, 0, true);

_checkbox.addEventListener(MouseEvent.CLICK, onCheckboxClick, false, 0, true);
_checkbox.addEventListener(MouseEvent.ROLL_OUT, onMouseRollOut, false, 0, true);
_checkbox.addEventListener(MouseEvent.ROLL_OVER, onMouseRollOver, false, 0, true);

addChild(_clickBG);
addChild(_checkbox);
addChild(_subject);
addChild(_sender);
addChild(_subject);
addChild(_dateSent);
addChild(_hoverImage);
}

public function init( mail:MailVO ):void
{
_mail = mail;
_sender.text = _mail.sender;
_subject.text = _mail.subject;
_dateSent.text = new Date(_mail.timeSent).toLocaleString();
_isRead = mail.isRead;
if(_isRead)
{
_bg.transform.colorTransform = new ColorTransform(0.5, 0.5, 0.5);
_sender.textColor = _subject.textColor = _dateSent.textColor = 0xbfbfbf;
}

}

private function onMouseRollOut( e:MouseEvent ):void
{
_hoverImage.visible = false;
}

private function onMouseRollOver( e:MouseEvent ):void
{
_hoverImage.visible = true;
}

private function onClick( e:MouseEvent ):void
{
```

```
if(e.target != _checkbox)
{
if(!_isRead)
isRead = true;
onClicked.dispatch(this);
}
}

private function onCheckboxClick( e:MouseEvent ):void
{
onSelectionChanged.dispatch(_checkbox.selected);
}

public function get timeSent():Number
{
return _mail.timeSent;
}

public function get mailKey():String
{
return _mail.key;
}

public function get mail():MailVO
{
return _mail;
}

public function get selected():Boolean
{
return _checkbox.selected;
}

public function set selected( v:Boolean ):void
{
_checkbox.selected = v;
}

public function set isRead( v:Boolean):void
{
_isRead = v;

if(v)
{
_bg.transform.colorTransform = new ColorTransform(0.5, 0.5, 0.5);
_sender.textColor = _subject.textColor = _dateSent.textColor = 0xbfbfbf;
}
}

public
```

```
function get isRead():Boolean
{
return _isRead;
}

override public function get height():Number
{
return _bg.height;
}

public function destroy():void
{

_clickBG.removeEventListener(MouseEvent.CLICK, onClick);
_clickBG.removeEventListener(MouseEvent.ROLL_OUT, onMouseRollOut);
_clickBG.removeEventListener(MouseEvent.ROLL_OVER, onMouseRollOver);
_clickBG.removeChild(_bg);
_clickBG = null;

_bg = null;
_hoverImage = null;

_checkbox.removeEventListener(MouseEvent.CLICK, onCheckboxClick);
_checkbox.removeEventListener(MouseEvent.ROLL_OUT, onMouseRollOut);
_checkbox.removeEventListener(MouseEvent.ROLL_OVER, onMouseRollOver);
_checkbox.destroy();
_checkbox = null;

_subject.destroy();
_subject = null;

_sender.destroy();
_sender = null;

_dateSent.destroy();
_dateSent = null;

_mail = null;

onClicked.removeAll();
onClicked = null;
}
}
}
```

File 768: igw\com\ui\hud\shared\mail\MailView.as

```
package com.ui.hud.shared.mail
{
import com.enum.ui.ButtonEnum;
import com.model.mail.MailVO;
import
```

```
com.model.player.CurrentUser;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.shared.ObjectPool;

public class MailView extends View
{
private var _bg:Bitmap;
private var _closeBtn:BitmapButton;
private var _windowFrame:Bitmap;

private var _messageFrame:ScaleBitmap;

private var _senderPortrait:ImageComponent;

private var _title:Label;
private var _subject:Label;
private var _body:Label;

private var _replyAllBtn:BitmapButton;
private var _replyBtn:BitmapButton;
private var _backArrowBtn:BitmapButton;
private var _deleteBtn:BitmapButton;
private var _replySymbolBtn:BitmapButton;

private var _deleteCallback:Function;

private var _mail:MailVO;

private var _message:String = 'CodeString.Mail.Message'; //MESSAGE
private var _reply:String = 'CodeString.Mail.Reply'; //REPLY

[PostConstruct]
override public function init():void
{
super.init();
```

```
var windowBGClass:Class = Class(getDefinitionByName('MailWindowMessageBMD'));
var messageFrameClass:Class = Class(getDefinitionByName('MailWindowOutlineBMD'));
_bg = new Bitmap(BitmapData(new windowBGClass()));

_windowFrame = new Bitmap(BitmapData(new messageFrameClass()));
_windowFrame.x = 17;
_windowFrame.y = 82;

_messageFrame = PanelFactory.getScaleBitmapPanel('MailMessageRowBMD', 650, 97, new
Rectangle(233, 16, 6, 6));
_messageFrame.x = 23;
_messageFrame.y = 102;

var portraitFrameClass:Class =
Class(getDefinitionByName('BattleLogLargePortraitFrameBMD'));
_senderPortrait = ObjectPool.get(ImageComponent);
_senderPortrait.init(2000, 2000);
_senderPortrait.x = _messageFrame.x + 5;
_senderPortrait.y = _messageFrame.y + 11;

_deleteBtn = ButtonFactory.getBitmapButton('MailTrashBtnUpBMD', 652, 45, '', 0,
'MailTrashBtnRollOverBMD', 'MailTrashBtnSelectedBMD', null, 'MailTrashBtnSelectedBMD');
addListener(_deleteBtn, MouseEvent.CLICK, onDeleteMail);

_replyBtn = ButtonFactory.getBitmapButton('MailReplyBtnUpBMD', 287, 374, _reply, 0xa9dcff,
'MailReplyBtnRollOverBMD', 'MailReplyBtnSelectedBMD', null, 'MailReplyBtnSelectedBMD');
addListener(_replyBtn, MouseEvent.CLICK, onReplyToMessage);

_replySymbolBtn = ButtonFactory.getBitmapButton('MailReplyIconUpBMD', 610, 45, '', 0,
'MailReplyIconRollOverBMD', 'MailReplyIconDownBMD', null, 'MailReplyIconDownBMD');
addListener(_replySymbolBtn, MouseEvent.CLICK, onReplyToMessage);

_backArrowBtn = ButtonFactory.getBitmapButton('MailBackArrowBtnUpBMD', 27, 59, '', 0,
'MailBackArrowBtnRollOverBMD', 'MailBackArrowBtnSelectedBMD', null,
'MailBackArrowBtnSelectedBMD');
addListener(_backArrowBtn, MouseEvent.CLICK, onBackArrowClick);

_replyAllBtn = UIFactory.getButton(ButtonEnum.REPLY_ALL);
addListener(_replyAllBtn, MouseEvent.CLICK, onReplyToAll);

_title = new Label(20, 0xf0f0f0, 150, 25, false);
_title.align = TextFormatAlign.LEFT;
_title.x = 25;
_title.y = 12;
_title.text = _message;

_subject = new Label(20, 0xf0f0f0, 600, 34, false);
_subject.align = TextFormatAlign.LEFT;
_subject.x
```

```
= 45;
_subject.y = 55;

_body = new Label(13, 0xf0f0f0, 559, 82, false, 1);
_body.align = TextFormatAlign.LEFT;
_body.multiline = true;
_body.x = 106;
_body.y = 108;

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

addChild(_bg);
addChild(_closeBtn);
addChild(_windowFrame);
addChild(_messageFrame);
addChild(_senderPortrait);
addChild(_title);
addChild(_subject);
addChild(_body);
addChild(_deleteBtn);
addChild(_replyBtn);
addChild(_replySymbolBtn);
addChild(_backArrowBtn);
// Removed ReplyAllBtn
//addChild(_replyAllBtn);

presenter.addOnMailDetailUpdatedListener(setUpView);

addEffects();
effectsIN();

}

public function setMail( mail:MailVO, deleteCallback:Function ):void
{
_mail = mail;
_deleteCallback = deleteCallback;

if (_mail.senderKey == null)
presenter.getMailDetails(_mail.key);
else
setUpView(_mail);
}

private function setUpView( mail:MailVO ):void
{
_mail = mail;

_subject.text = _mail.subject;
_title.text
```

```
= _mail.sender.toUpperCase();
_body.text = _mail.body;

if (mail.sendersRace != '')
presenter.loadPortraitSmall(mail.sendersRace, _senderPortrait.onImageLoaded);

_replyBtn.visible = (_mail.senderKey != '');
_replySymbolBtn.visible = (_mail.senderKey != '');
_replyAllBtn.visible = (_mail.allianceKey != '' && _mail.allianceKey == CurrentUser.alliance);

var xPos:Number = 590;

if (_replyAllBtn.visible)
{
_replyAllBtn.x = xPos;
_replyAllBtn.y = 47;

xPos += _replyAllBtn.width + 10;
}

_replySymbolBtn.x = xPos;
xPos += _replySymbolBtn.width + 10;
_deleteBtn.x = xPos;

}

private function onDeleteMail( e:MouseEvent ):void
{
_deleteCallback(_mail.key);
destroy();
}

private function onReplyToMessage( e:MouseEvent ):void
{
var newMailView:NewMailView = NewMailView(_viewFactory.createView(NewMailView));
newMailView.setMessageInfo(_mail.sender, _mail.senderKey, _mail.subject);
_viewFactory.notify(newMailView);
}

private function onReplyToAll( e:MouseEvent ):void
{
// Removed ReplyToAll Button
/*
var newMailView:NewMailView = NewMailView(_viewFactory.createView(NewMailView));
newMailView.setMessageInfo('Alliance', '', _mail.subject);
_viewFactory.notify(newMailView);
*/
}

private function onBackArrowClick( e:MouseEvent ):void
{
```

```
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
presenter.removeOnMailDetailUpdatedListener(setUpView);
super.destroy();

_bg = null;
_closeBtn.destroy();
_closeBtn = null;
_windowFrame = null;
_messageFrame = null;

_title.destroy();
_title = null;

_subject.destroy();
_subject = null;

_body.destroy();
_body = null;

_replyBtn.destroy();
_replyBtn = null;

_backArrowBtn.destroy();
_backArrowBtn = null;

_deleteBtn.destroy();
_deleteBtn = null;

_replySymbolBtn.destroy();
_replySymbolBtn = null;

_deleteCallback = null;

_mail = null;
}
}
}

-----------------------------------
```

File 769: igw\com\ui\hud\shared\mail\NewMailView.as

```
package com.ui.hud.shared.mail
{
import com.presenter.shared.IUIPresenter;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

public class NewMailView extends View
{
private var _name:String;
private var _playerID:String;
private var _subjectText:String;

private var _bg:Bitmap;
private var _closeBtn:BitmapButton;
private var _title:Label;
private var _to:Label;
private var _subject:Label;
private var _body:Label;

private var _sendBtn:BitmapButton;

private var _newMessage:String = 'CodeString.Mail.NewMessage'; //NEW MESSAGE
private var _subjectLocText:String = 'CodeString.Mail.Subject'; //Subject
private var _reply:String = 'CodeString.Mail.ReplyText'; //RE: [[1]]
private var _send:String = 'CodeString.Mail.Send'; //SEND
private var _bodyText:String = 'CodeString.Mail.Body'; //Write Message Here


[PostConstruct]
override public function init():void
{
super.init();

var windowBGClass:Class = Class(getDefinitionByName('MailNewMessageWindowBGBMD'));
_bg = new Bitmap(BitmapData(new windowBGClass()));

_title = new Label(20, 0xf0f0f0, 150, 25, true);
_title.align = TextFormatAlign.LEFT;
_title.x = 25;
_title.y = 12;
_title.text
```

```
= _newMessage;

_to = new Label(18, 0xa9dcff, 468, 25, false);
_to.align = TextFormatAlign.LEFT;
_to.x = 37;
_to.y = 55;
_to.text = _name;

_subject = new Label(18, 0xa9dcff, 468, 25, true);
_subject.align = TextFormatAlign.LEFT;
_subject.x = 37;
_subject.y = 82;
if (_subjectText == '')
{
_subject.text = _subjectLocText;
_subject.maxChars = 45;
_subject.allowInput = true;
_subject.clearOnFocusIn = true;
_subject.letterSpacing = .8;
_subject.addLabelColor(0xbdfefd, 0x000000);
} else
{
if (_subjectText.indexOf('RE:') == -1)
_subject.setTextWithTokens(_reply, {'[[String.Subject]]':_subjectText});
else
{
_subject.useLocalization = false
_subject.text = _subjectText;
}
}

_body = new Label(18, 0xa9dcff, 468, 106, true);
_body.align = TextFormatAlign.LEFT;
_body.text = _bodyText;
_body.x = 37;
_body.y = 118;
_body.maxChars = 250;
_body.multiline = true;
_body.allowInput = true;
_body.clearOnFocusIn = true;
_body.letterSpacing = .8;
_body.addLabelColor(0xbdfefd, 0x000000);

_sendBtn = ButtonFactory.getBitmapButton('MailReplyBtnUpBMD', 370, 235, _send, 0xa9dcff,
'MailReplyBtnRollOverBMD', 'MailReplyBtnSelectedBMD', null, 'MailReplyBtnSelectedBMD');
addListener(_sendBtn, MouseEvent.CLICK, onSendMail);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

addChild(_bg);
```

```
addChild(_closeBtn);
addChild(_title);
addChild(_to);
addChild(_subject);
addChild(_body);
addChild(_sendBtn);

addEffects();
effectsIN();
}

private function onSendMail( e:MouseEvent ):void
{
if (_subject.text != _subjectLocText && _body.text != _bodyText)
{
if (_playerID != '')
presenter.sendMailMessage(_playerID, _subject.text, _body.text);
else
presenter.sendAllianceMailMessage(_subject.text, _body.text);

destroy();
}
}

public function setMessageInfo( name:String, playerID:String = '', subjectText:String = '' ):void
{
_name = name;
_playerID = playerID;
_subjectText = subjectText;
}

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
super.destroy();

_bg = null;
_closeBtn.destroy();
_closeBtn = null;

_title.destroy();
_title = null;

_to.destroy();
_to = null;

_subject.destroy();
```

```
_subject = null;

_body.destroy();
_body = null;

_sendBtn.destroy();
_sendBtn = null;
}
}
}
```

File 770: igw\com\ui\hud\starbase\StarbaseBaseView.as

```
package com.ui.hud.starbase
{
import com.presenter.starbase.IStarbasePresenter;
import com.ui.core.View;

import org.parade.enum.ViewEnum;

public class StarbaseBaseView extends View
{
[PostConstruct]
override public function init():void
{
super.init();
presenter.addCleanupListener(destroy);
}

[Inject]
public function set presenter( value:IStarbasePresenter ):void { _presenter = value; }
public function get presenter():IStarbasePresenter { return IStarbasePresenter(_presenter); }

override public function get type():String { return ViewEnum.UI; }

override public function get screenshotBlocker():Boolean {return true;}

override public function destroy():void
{
presenter.removeCleanupListener(destroy);
super.destroy();
}
}
}
```

File 771: igw\com\ui\hud\starbase\StarbaseView.as

```
package com.ui.hud.starbase
{
import
```

```
com.Application;
import com.enum.PositionEnum;
import com.enum.ToastEnum;
import com.enum.TypeEnum;
import com.enum.server.AllianceRankEnum;
import com.enum.server.AllianceResponseEnum;
import com.event.TransactionEvent;
import com.game.entity.components.shared.Detail;
import com.game.entity.components.starbase.Building;
import com.game.entity.components.starbase.Platform;
import com.game.entity.components.starbase.State;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.transaction.TransactionVO;
import com.presenter.shared.IUIPresenter;
import com.service.ExternalInterfaceAPI;
import com.service.language.Localization;
import com.ui.core.component.contextmenu.ContextMenu;
import com.ui.core.effects.EffectFactory;
import com.ui.modal.building.RefitBuildingView;
import com.ui.modal.building.RepairBaseView;
import com.ui.modal.construction.ConstructionInfoView;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.dock.DockView;
import com.ui.modal.shipyard.ShipyardView;
import com.ui.modal.store.StoreView;

import flash.display.Stage;

import org.ash.core.Entity;
import org.parade.util.DeviceMetrics;
import org.shared.ObjectPool;

public class StarbaseView extends StarbaseBaseView
{
private var _stage:Stage;
private var _uiPresenter:IUIPresenter;

private const MIN_X_POS:Number = 650;
private const EXIT_MIN_X_POS:Number = 67;

private var _build:String = 'CodeString.Controls.Build'; //BUILD
private var _fleets:String = 'CodeString.Controls.Fleets'; //FLEETS
private var _shipyard:String = 'CodeString.Controls.Shipyard'; //SHIPYARD
private var _research:String = 'CodeString.Controls.Research'; //RESEARCH
private var _store:String = 'CodeString.Controls.Store'; //STORE

private var _contextMenuStarbaseText:String = 'CodeString.ContextMenu.Starbase.Move';
//Move
private
```

```
var _contextMenuRecycleStructureText:String =
'CodeString.ContextMenu.Starbase.RecycleStructure'; //Recycle Structure
private var _contextMenuRepairText:String = 'CodeString.ContextMenu.Starbase.Repair';
//Repair
private var _contextMenuBuildShipText:String = 'CodeString.ContextMenu.Starbase.BuildShip';
//Build Ship
private var _contextMenuViewFleetsText:String =
'CodeString.ContextMenu.Starbase.ViewFleets'; //View Fleets
private var _contextMenuRefitText:String = 'CodeString.ContextMenu.Starbase.Refit'; //Refit
private var _contextMenuShipDesignerText:String =
'CodeString.ContextMenu.Starbase.ShipDesigner'; //Ship Designer
private var _contextMenuResearchDefensesText:String =
'CodeString.ContextMenu.Starbase.ResearchDefenses'; //Research Defenses

private var _contextMenuResearchWeaponsText:String =
'CodeString.ContextMenu.Starbase.ResearchWeapons'; //Research Weapons
private var _contextMenuResearchTechText:String =
'CodeString.ContextMenu.Starbase.ResearchTech'; //Research Tech
private var _contextMenuUpgradeText:String = 'CodeString.ContextMenu.Starbase.Upgrade';
//Upgrade
private var _contextMenuDetailsText:String = 'CodeString.ContextMenu.Starbase.Details';
//Details
private var _contextMenuSpeedUpText:String = 'CodeString.ContextMenu.Starbase.SpeedUp';
//Speed Up
private var _contextMenuCancelBuildText:String =
'CodeString.ContextMenu.Starbase.CancelBuild'; //Cancel Build
private var _contextMenuCancelRepairText:String =
'CodeString.ContextMenu.Starbase.CancelRepair'; //Cancel Repair
private var _contextMenuCancelResearchText:String =
'CodeString.ContextMenu.Starbase.CancelResearch'; //Cancel Research
private var _contextMenuCancelUpgradeText:String =
'CodeString.ContextMenu.Starbase.CancelUpgrade'; //Cancel Upgrade

private var _alertFleetBattleTitleText:String = 'CodeString.Alert.FleetBattle.Title'; //Fleet Battle!
private var _alertFleetBattleBodyText:String = 'CodeString.Alert.FleetBattle.Body'; //You are
currently engaged in a fleet battle.
private var _alertBaseBattleTitleText:String = 'CodeString.Alert.BaseBattle.Title'; //Base Battle!
private var _alertBaseBattleBodyText:String = 'CodeString.Alert.BaseBattle.Body'; //You are
currently engaged in a base battle.
private var _alertViewBtnText:String = 'CodeString.Alert.BaseBattle.ViewBtn'; //View
private var _alertDontViewBtnText:String = 'CodeString.Alert.FleetBattle.DontViewBtn'; //Dont
View

private var _sharedLevel:String = 'CodeString.Shared.Level'; //Level [[Number.Level]]

private var _toastAllianceRemoved:String = 'CodeString.Toast.AllianceRemoved'; //You have
been removed from your alliance.
private var _toastAllianceBadName:String = 'CodeString.Toast.AllianceBadName'; //Alliance
creation failed bad name.
private var _toastAllianceNameInUse:String = 'CodeString.Toast.AllianceNameInUse'; //Alliance
creation
```

```
failed name in use.
private var _toastAllianceFailedOffline:String = 'CodeString.Toast.AllianceInviteFailedOffline';
//Alliance invite failed target offline.
private var _toastAllianceFailedIgnored:String = 'CodeString.Toast.AllianceInviteFailedIgnored';
//Alliance invite failed target is ignoring invites.
private var _toastAllianceAlreadyInAnAlliance:String =
'CodeString.Toast.AllianceInviteFailedAlreadyInAnAlliance'; //Alliance invite failed target is
already in an alliance.
private var _toastAllianceTooManyAlready:String =
'CodeString.Toast.AllianceJoinFailedTooManyAlready'; //Alliance join failed too many players in
alliance.
private var _toastAllianceNoLongerExists:String =
'CodeString.Toast.AllianceJoinFailedNoLongerExists'; //Alliance join failed it no longer exists
private var _toastAllianceInviteRecieved:String = 'CodeString.Toast.AllianceInviteRecieved';
//You have been invited to join an alliance.
private var _toastAllianceJoined:String = 'CodeString.Toast.AllianceJoined'; //You have joined
an alliance.


[PostConstruct]
override public function init():void
{
super.init();
presenter.addBaseInteractionListener(popContextMenuFromEntity);
presenter.showBuildings();
_uiPresenter && _uiPresenter.highfive();

x = DeviceMetrics.WIDTH_PIXELS * 0.5;

if (x < MIN_X_POS)
x = MIN_X_POS;

presenter.addOnGenericAllianceMessageRecievedListener(onGenericAllianceMessage);

addHitArea();
addEffects();
effectsIN();
onResize();
}


//==========================================================================
//**************************************************************************************************
// ENTITY INTERACTION
//**************************************************************************************************

//==========================================================================

private function popContextMenuFromEntity( x:int, y:int, baseEntity:Entity ):void
{
var
```

```
contextMenu:ContextMenu = ObjectPool.get(ContextMenu);
var buildingState:State = State(baseEntity.get(State));
var detail:Detail = baseEntity.get(Detail);
var buildingVO:BuildingVO = (baseEntity.has(Building)) ?
Building(baseEntity.get(Building)).buildingVO : Platform(baseEntity.get(Platform)).buildingVO;
var buildingName:String = presenter.getEntityName(buildingVO.asset);
var level:String = Localization.instance.getStringWithTokens(_sharedLevel,
{'[[Number.Level]]':buildingVO.level});

contextMenu.setup(buildingName, x, y, 150, _stage.stageWidth, _stage.stageHeight, level);
if (baseEntity.has(Platform))
{
contextMenu.addContextMenuChoice(_contextMenuStarbaseText, presenter.moveEntity, []);
if (buildingVO.prototype.getValue("canBeRecycled") == true)
//contextMenu.addContextMenuChoice(_contextMenuRecycleStructureText, showRecycleView,
[baseEntity]);
contextMenu.addContextMenuChoice(_contextMenuDetailsText, showUpgradeView,
[baseEntity]);
} else
{
if (buildingVO.currentHealth < 1 && (!buildingState || buildingState.type !=
TransactionEvent.STARBASE_REPAIR_BASE))
{
contextMenu.addContextMenuChoice(_contextMenuRepairText, showBaseRepair, []);
contextMenu.addContextMenuChoice(_contextMenuStarbaseText, presenter.moveEntity, []);
} else if (!buildingState)
{
switch (detail.type)
{
case TypeEnum.CONSTRUCTION_BAY:
contextMenu.addContextMenuChoice(_contextMenuBuildShipText, showShipyard, []);
break;
case TypeEnum.DOCK:
contextMenu.addContextMenuChoice(_contextMenuViewFleetsText, showDocks, []);
break;
case TypeEnum.SHIPYARD:
contextMenu.addContextMenuChoice(_contextMenuShipDesignerText, enterResearchView,
[baseEntity]);
break;
case TypeEnum.DEFENSE_DESIGN:
contextMenu.addContextMenuChoice(_contextMenuResearchDefensesText,
enterResearchView, [baseEntity]);
break;
case TypeEnum.WEAPONS_FACILITY:
contextMenu.addContextMenuChoice(_contextMenuResearchWeaponsText,
enterResearchView, [baseEntity]);
break;
case TypeEnum.ADVANCED_TECH:
contextMenu.addContextMenuChoice(_contextMenuResearchTechText, enterResearchView,
[baseEntity]);
break;
```

```
case TypeEnum.POINT_DEFENSE_PLATFORM:
case TypeEnum.SHIELD_GENERATOR:
contextMenu.addContextMenuChoice(_contextMenuRefitText, showRefitView, [baseEntity]);
break;
}
var upgradePrototype:IPrototype = presenter.getBuildingUpgrade(buildingVO);
if (upgradePrototype)
contextMenu.addContextMenuChoice(_contextMenuUpgradeText, showUpgradeView,
[baseEntity]);
else
contextMenu.addContextMenuChoice(_contextMenuDetailsText, showUpgradeView,
[baseEntity]);
contextMenu.addContextMenuChoice(_contextMenuStarbaseText, presenter.moveEntity, []);
} else
{
if (buildingState.type == TransactionEvent.STARBASE_REPAIR_BASE)
{
contextMenu.addContextMenuChoice(_contextMenuSpeedUpText, showTransactionView,
[buildingState.transaction]);
contextMenu.addContextMenuChoice(_contextMenuStarbaseText, presenter.moveEntity, []);
} else
{
if (buildingState.type == TransactionEvent.STARBASE_REFIT_BUILDING)
contextMenu.addContextMenuChoice(_contextMenuRefitText, showRefitView, [baseEntity]);
//PR: Adding the speed up option back in for now so that the fte isn't broken
contextMenu.addContextMenuChoice(_contextMenuSpeedUpText, showTransactionView,
[buildingState.transaction]);
if (buildingState.type != TransactionEvent.STARBASE_BUILDING_UPGRADE)
contextMenu.addContextMenuChoice(_contextMenuUpgradeText, showUpgradeView,
[baseEntity]);
contextMenu.addContextMenuChoice(_contextMenuStarbaseText, presenter.moveEntity, []);
switch (buildingState.type)
{
case TransactionEvent.STARBASE_BUILDING_BUILD:
contextMenu.addContextMenuChoice(_contextMenuCancelBuildText,
presenter.cancelTransaction, [buildingState.transaction]);
break;
case TransactionEvent.STARBASE_RESEARCH:
contextMenu.addContextMenuChoice(_contextMenuCancelResearchText,
presenter.cancelTransaction, [buildingState.transaction]);
break;
case TransactionEvent.STARBASE_BUILDING_UPGRADE:
contextMenu.addContextMenuChoice(_contextMenuCancelUpgradeText,
presenter.cancelTransaction, [buildingState.transaction]);
break;
}
}
}
}

_viewFactory.notify(contextMenu);
```

```
}

private function showBaseRepair():void
{
showView(RepairBaseView);
}

private function enterResearchView( baseEntity:Entity ):void
{
var view:ConstructionView = ConstructionView(_viewFactory.createView(ConstructionView));
view.openOn(ConstructionView.RESEARCH,
Building(baseEntity.get(Building)).buildingVO.asset, null);
_viewFactory.notify(view);
}

private function showRefitView( baseEntity:Entity ):void
{
var nShieldView:RefitBuildingView =
RefitBuildingView(_viewFactory.createView(RefitBuildingView));
nShieldView.buildingVO = presenter.getBuildingVO(baseEntity.id);
_viewFactory.notify(nShieldView);
}

private function showUpgradeView( baseEntity:Entity ):void
{
var view:ConstructionInfoView =
ConstructionInfoView(_viewFactory.createView(ConstructionInfoView));
view.setup(ConstructionView.BUILD, presenter.getBuildingVO(baseEntity.id));
_viewFactory.notify(view);
}

private function showTransactionView( transaction:TransactionVO ):void
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
nStoreView.setSelectedTransaction(transaction);
}

private function showDocks():void { showView(DockView); }
private function showShipyard():void { showView(ShipyardView); }


//==========================================================================
//**************************************************************************************************
// CONTROLS
//**************************************************************************************************

//==========================================================================

private
```

```
function onResize():void
{
this.scaleX = this.scaleY = Application.SCALE;
x = DeviceMetrics.WIDTH_PIXELS * 0.5;

if (x < MIN_X_POS)
x = MIN_X_POS;
}

override protected function addEffects():void
{
_effects.addEffect(EffectFactory.repositionEffect(PositionEnum.CENTER, PositionEnum.TOP,
onResize));
}

private function onGenericAllianceMessage( messageEnum:int, allianceKey:String ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.SET_SUCCESS:
break;
case AllianceResponseEnum.KICKED:
case AllianceResponseEnum.LEFT:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceRemoved);
CurrentUser.alliance = '';
CurrentUser.allianceName = '';
CurrentUser.allianceRank = AllianceRankEnum.UNAFFILIATED;
CurrentUser.isAllianceOpen = false;
break;
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_BADNAME:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceBadName);
break;
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_NAMEINUSE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceNameInUse);
break;
case AllianceResponseEnum.INVITE_FAILED_OFFLINE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceFailedOffline);
break;
case AllianceResponseEnum.INVITE_FAILED_IGNORED:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceFailedIgnored);
break;
case AllianceResponseEnum.INVITE_FAILED_INALLIANCE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceAlreadyInAnAlliance);
break;
case AllianceResponseEnum.JOIN_FAILED_TOOMANYPLAYERS:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceTooManyAlready);
break;
case AllianceResponseEnum.JOIN_FAILED_NOALLIANCE:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceNoLongerExists);
break;
case
```

```
AllianceResponseEnum.INVITED:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceInviteRecieved);
break;
case AllianceResponseEnum.JOINED:
showToast(ToastEnum.ALLIANCE, null, _toastAllianceJoined);
ExternalInterfaceAPI.shareAllianceJoin();
break;
}

}

[Inject]
public function set stage( value:Stage ):void { _stage = value; }
[Inject]
public function set uiPresenter( v:IUIPresenter ):void { _uiPresenter = v; }

override public function destroy():void
{
presenter.removeOnGenericAllianceMessageRecievedListener(onGenericAllianceMessage);

_uiPresenter && _uiPresenter.shun();
_uiPresenter = null;

super.destroy();
_stage = null;
}
}
}

-------------------------------------
File 772: igw\com\ui\modal\ButtonFactory.as
package com.ui.modal
{
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.button.ButtonBase;

import flash.display.BitmapData;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

public class ButtonFactory
{
private static const _bmdStore:Dictionary = new Dictionary();

public static function getBitmapButton( upName:String, dx:Number, dy:Number, text:String = '',
textColor:uint = 0, overName:String = null, downName:String = null,
disabledName:String = null, selectName:String = null, horzMargin:Number = 0,
vertMargin:Number = 0 ):BitmapButton
{
var button:BitmapButton = new BitmapButton();
button.x
```

```
= dx;
button.y = dy;
var upSkin:BitmapData = getBMD(upName);
var overSkin:BitmapData = getBMD(overName);
var downSkin:BitmapData = getBMD(downName);
var disabledSkin:BitmapData = getBMD(disabledName);
var selectSkin:BitmapData = getBMD(selectName);
button.init(upSkin, overSkin, downSkin, disabledSkin, selectSkin);
applyText(button, text, textColor);
button.horzTxtMargin = horzMargin;
button.vertTxtMargin = vertMargin;
return button;
}

public static function getCloseButton( x:Number, y:Number ):BitmapButton { return
getBitmapButton('CloseBtnUpBMD', x, y, '', 0, 'CloseBtnRollOverBMD', 'CloseBtnDownBMD'); }

private static function applyText( button:ButtonBase, text:String, textColor:uint ):void
{
if (text != null && text != '')
{
button.text = text;
button.textColor = textColor;
}
}

public static function getBMD( name:String ):BitmapData
{
if (name == null || name == '')
return null;
if (!_bmdStore[name])
{
var bmdClass:Class = Class(getDefinitionByName(name));
var bmd:BitmapData = BitmapData(new bmdClass());
_bmdStore[name] = bmd;
}
return _bmdStore[name];
}
}
}

-----------------------------------
File 773: igw\com\ui\modal\PanelFactory.as
package com.ui.modal
{
import com.ui.core.ScaleBitmap;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.geom.Rectangle;
import
```

```
flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

public class PanelFactory
{
private static const _bmdStore:Dictionary = new Dictionary();

public static function getScaleBitmapPanel( name:String, width:Number = -1, height:Number =
-1, scale9Rect:Rectangle = null ):ScaleBitmap
{
var panel:ScaleBitmap = new ScaleBitmap();
if (!_bmdStore[name])
getBitmapData(name);
panel.bitmapData = _bmdStore[name];
if (scale9Rect)
panel.scale9Grid = scale9Rect;
if (width > -1)
panel.width = width;
if (height > -1)
panel.height = height;
return panel;
}

public static function getPanel( name:String ):Bitmap
{
var panel:Bitmap = new Bitmap();
if (!_bmdStore[name])
getBitmapData(name);
panel.bitmapData = _bmdStore[name];
return panel;
}

public static function getBitmapData( name:String ):BitmapData
{
var bmdClass:Class = Class(getDefinitionByName(name));
var bmd:BitmapData = BitmapData(new bmdClass());
_bmdStore[name] = bmd;
return bmd;
}
}
}

----------------------------------
File 774: igw\com\ui\modal\achievements\AchievementDisplay.as
package com.ui.modal.achievements
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.achievements.AchievementVO;
import
```

```
com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

import org.adobe.utils.StringUtil;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class AchievementDisplay extends Sprite
{
public var onLoadImage:Signal;
public var onClaimReward:Signal;
public var maxCredits:Function;
public var getScore:Function;

private var _bg:Sprite;

private var _starUnlit:BitmapData
private var _starLit:BitmapData;

private var _scoreProgressBar:ProgressBar;

private var _description:Label;
private var _rewardsLabel:Label;
private var _scoreLabel:Label;

public var _claimBtn:BitmapButton;

private var _rewards:Vector.<AchievementReward>;

private var _achievements:Dictionary;
private var _achievementsAsset:Dictionary;
private var _achievementProgress:Dictionary;

private var _achievementImage:ImageComponent;

private var _category:String;
private var _currentRank:uint;
private
```

```
var _nextRankKey:String;
private var _sort:Number;
private var _maxScore:Number;
private var _scoreKey:String;

private var _stars:Vector.<Bitmap>;

private var _rewardText:String = 'CodeString.Achievements.Rewards'; //REWARDS:
private var _claimBtnText:String = 'CodeString.Achievement.ClaimBtn'; //CLAIM
private var _outOfString:String = 'CodeString.Shared.OutOf';
//[[Number.MinValue]]/[[Number.MaxValue]]

public function AchievementDisplay( category:String )
{
super();

onLoadImage = new Signal(String, Function);
onClaimReward = new Signal(String);

_rewards = new Vector.<AchievementReward>;
_achievements = new Dictionary;
_achievementsAsset = new Dictionary;
_achievementProgress = new Dictionary;

_starUnlit = UIFactory.getBitmapData('IconStarInactiveBMD');
_starLit = UIFactory.getBitmapData('IconStarActiveBMD');

_bg = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_INNER,
PanelEnum.HEADER_NOTCHED, 582, 137, 30, 0, 0, 'Achievement');

_stars = new Vector.<Bitmap>;

_description = new Label(18, 0xf0f0f0, 440, 35);
_description.multiline = true;
_description.constrictTextToSize = false;
_description.align = TextFormatAlign.LEFT;
_description.x = 134;
_description.y = 63;

_rewardsLabel = new Label(22, 0xf0f0f0, 125, 25);
_rewardsLabel.constrictTextToSize = false;
_rewardsLabel.align = TextFormatAlign.LEFT;
_rewardsLabel.x = 132;
_rewardsLabel.y = 130;
_rewardsLabel.text = _rewardText;

_achievementImage = ObjectPool.get(ImageComponent);
_achievementImage.init(120, 120);
_achievementImage.x = 9;
_achievementImage.y = 39;

_claimBtn
```

```
= UIFactory.getButton(ButtonEnum.GREEN_A, 100, 30, 0, 0, _claimBtnText, LabelEnum.H1);
_claimBtn.x = _bg.width - _claimBtn.width - 19;
_claimBtn.y = 34;
_claimBtn.visible = false;
_claimBtn.addEventListener(MouseEvent.CLICK, onClickClaimReward, false, 0, true);

_category = category;

_currentRank = 1;
_nextRankKey = _category + '_' + _currentRank;

addChild(_bg);
addChild(_description);
addChild(_rewardsLabel);
addChild(_claimBtn);
addChild(_achievementImage);
}

public function setUpStars( maximumRank:int ):void
{
for(var i:int=0; i<maximumRank; i++){
var star:Bitmap = new Bitmap(_starUnlit);
star.x = 132 + 34 * i;
star.y = 33;
addChild(star);
_stars.push(star);
}
}

public function addAchievement( achievement:IPrototype, assetVO:AssetVO ):void
{
_achievements[achievement.name] = achievement;
_achievementsAsset[achievement.name] = assetVO;

if (achievement.getValue('rank') == 1)
updateAchievementDisplay(_nextRankKey);
}

public function addAchievementProgress( achievementProgress:AchievementVO ):void
{
_achievementProgress[achievementProgress.achievementPrototype] = achievementProgress;

if (achievementProgress.achievementPrototype in _achievements)
{
if (!achievementProgress.claimedFlag && !_claimBtn.visible)
_claimBtn.visible = true;

var achievement:IPrototype = _achievements[achievementProgress.achievementPrototype];
if (achievementProgress.claimedFlag)
updateRank(achievement.getValue('rank'));
else
```

```
cleanUpScoreBar();
}
}

private function updateRank( rank:uint ):void
{
for (var i:int = 0; i < rank; i++)
{
if(i < _stars.length)
{
_stars[i].bitmapData = _starLit;
}
}

if (rank >= _currentRank)
{
_currentRank = rank;
_nextRankKey = _category + '_' + (rank + 1);

if (_nextRankKey in _achievementsAsset)
updateAchievementDisplay(_nextRankKey);
else
{
_nextRankKey = _category + '_' + rank;
_rewardsLabel.visible = false;
_description.visible = false;
removeRewards();
}
}
}

private function onClickClaimReward( e:MouseEvent ):void
{
var rankKey:String;
var achievementProgress:AchievementVO;
var alreadyClaimed:Boolean;
for (var i:uint = 1; i <= _stars.length; ++i)
{
rankKey = _category + '_' + i;
if (rankKey in _achievementProgress)
{
achievementProgress = _achievementProgress[rankKey];
if (!achievementProgress.claimedFlag)
{
if (!alreadyClaimed)
{
var achievement:IPrototype = _achievements[achievementProgress.achievementPrototype];
achievementProgress.claimedFlag = alreadyClaimed = true;
_claimBtn.visible = false;
_achievementProgress[rankKey]
```

```
= achievementProgress;

updateRank(achievement.getValue('rank'));

if (onClaimReward != null)
onClaimReward.dispatch(achievementProgress.key);

} else
{
_claimBtn.visible = true;
break;
}
}
}
}
}

private function updateAchievementDisplay( key:String ):void
{
var achievement:IPrototype;
var assetVO:AssetVO;
var achievementProgress:AchievementVO;

if (key in _achievements)
achievement = _achievements[key];

if (key in _achievementsAsset)
assetVO = _achievementsAsset[key];

if (key in _achievementProgress)
achievementProgress = _achievementProgress[key];

_scoreKey = '';
removeRewards();

if (onLoadImage && assetVO)
onLoadImage.dispatch(assetVO.smallImage, _achievementImage.onImageLoaded);

if (_bg && assetVO)
Label(_bg.getChildAt(2)).text = assetVO.visibleName;

if (_description && assetVO)
_description.text = assetVO.descriptionText;

if (achievement)
{
_sort = achievement.getValue('sort');
if (achievementProgress == null)
{
_scoreKey = achievement.getValue('scoreKey');
_maxScore
```

```
= achievement.getValue('scoreRequired');
}

if (scoreKey == '' || scoreKey == null)
cleanUpScoreBar();
}

if (_rewards && achievement)
{
var rewardAmount:Number = achievement.getValue('hardCurrencyReward');
if (rewardAmount > 0)
addReward(rewardAmount, AchievementReward.REWARD_PALLADIUM);

rewardAmount = achievement.getValue('experienceReward');
if (rewardAmount > 0)
addReward(rewardAmount, AchievementReward.REWARD_EXP);

rewardAmount = achievement.getValue('creditsReward');
if (rewardAmount > 0)
{
if (maxCredits != null)
rewardAmount *= maxCredits();
addReward(rewardAmount, AchievementReward.REWARD_CREDITS);
}

if (achievement.getValue('blueprintReward') == true)
{
var bpParts:int = achievement.getValue('blueprintRewardParts');
if(bpParts == 0) bpParts = 1;
addReward(bpParts, AchievementReward.BLUEPRINT);
}

layoutRewards();
}
}

private function addReward( amount:int, type:int ):void
{
var reward:AchievementReward = new AchievementReward(amount, type);
addChild(reward);
_rewards.push(reward)
}

private function layoutRewards():void
{
var xPos:Number = _rewardsLabel.x + _rewardsLabel.textWidth + 4;
var yPos:Number = 130;

var len:uint = _rewards.length;
var currentReward:AchievementReward;
for
```

```
(var i:uint = 0; i < len; ++i)
{
currentReward = _rewards[i];
currentReward.x = xPos;
currentReward.y = yPos;

xPos += currentReward.width + 10;
}
}

private function removeRewards():void
{
if (_rewards)
{
var len:uint = _rewards.length;
var currentReward:AchievementReward;
for (var i:uint = 0; i < len; ++i)
{
currentReward = _rewards[i];
removeChild(currentReward);
currentReward.destroy();
currentReward = null;
}
_rewards.length = 0;
}
}

public function setScore( v:Number ):void
{
var achievementProgress:AchievementVO;
if (_nextRankKey in _achievementProgress)
achievementProgress = _achievementProgress[_nextRankKey];

if (achievementProgress == null || achievementProgress.claimedFlag)
setUpScoreBar(v);
}

private function setUpScoreBar( current:Number ):void
{
if (current > _maxScore)
current = _maxScore;

if (_scoreProgressBar == null)
{
_scoreProgressBar = UIFactory.getProgressBar(UIFactory.getPanel(PanelEnum.STATBAR,
428, 16), UIFactory.getPanel(PanelEnum.STATBAR_CONTAINER, 436, 24), 0, _maxScore,
current, 135, 102);
addChild(_scoreProgressBar);
}
_scoreProgressBar.setMinMax(0, _maxScore);
_scoreProgressBar.amount
```

```
= current;

if (_scoreLabel == null)
{
_scoreLabel = new Label(14, 0xf0f0f0, 436, 25, true, 1);
_scoreLabel.letterSpacing = 1.5;
_scoreLabel.align = TextFormatAlign.CENTER;
_scoreLabel.y = _scoreProgressBar.y;
_scoreLabel.x = _scoreProgressBar.x;
_scoreLabel.constrictTextToSize = false;
addChild(_scoreLabel);
}
_scoreLabel.setTextWithTokens(_outOfString,
{'[[Number.MinValue]]':StringUtil.commaFormatNumber(current),
'[[Number.MaxValue]]':StringUtil.commaFormatNumber(_maxScore)});
}

private function cleanUpScoreBar():void
{
if (_scoreProgressBar)
{
removeChild(_scoreProgressBar);
_scoreProgressBar.destroy();
}

_scoreProgressBar = null;

if (_scoreLabel)
{
removeChild(_scoreLabel);
_scoreLabel.destroy();
}

_scoreLabel = null;
}

public function get sort():Number { return _sort; }
public function get scoreKey():String { return _scoreKey; }

public function achievementsCount():int
{
var n:int = 0;
for (var key:* in _achievements) {
n++;
}
return n;
}

public function destroy():void
{
removeRewards();
```

```
if (onLoadImage)
onLoadImage.removeAll();

onLoadImage = null;

if (onClaimReward)
onClaimReward.removeAll();

onClaimReward = null;

_bg = null;

_stars = null;

_starUnlit = null;
_starLit = null;

cleanUpScoreBar();

if (_achievementImage)
ObjectPool.give(_achievementImage)

_achievementImage = null

if (_description)
_description.destroy();

_description = null;

if (_rewardsLabel)
_rewardsLabel.destroy();

_rewardsLabel = null;

if (_claimBtn)
{
_claimBtn.removeEventListener(MouseEvent.CLICK, onClickClaimReward);
_claimBtn.destroy();
}

_claimBtn = null;

_achievements = null;
_achievementsAsset = null;
_achievementProgress = null;
}
}
}

-----------------------------------
```

File 775: igw\com\ui\modal\achievements\AchievementReward.as

```
package com.ui.modal.achievements
{
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.label.Label;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;

public class AchievementReward extends Sprite
{

public static var REWARD_PALLADIUM:uint = 0;
public static var REWARD_EXP:uint = 1;
public static var REWARD_CREDITS:uint = 2;
public static var BLUEPRINT:uint = 3;

private var _bg:ScaleBitmap;
private var _rewardSymbol:Bitmap;

private var _type:uint;

private var _reward:Label;
private var _rewardValue:Label;

private var _value:Number;

private var _exp:String = 'CodeString.Achievement.Exp'; //EXP:
private var _blueprintPiece:String = 'CodeString.Achievement.BlueprintPiece'; //BLUEPRINT
PIECE:

public function AchievementReward( value:Number, type:uint )
{
_type = type;
_value = value;

_bg = UIFactory.getScaleBitmap(PanelEnum.STATBAR_CONTAINER);
_bg.width = 130;
_bg.height = 24;

_rewardValue = new Label(20, 0xf0f0f0);
_rewardValue.constrictTextToSize = false;
_rewardValue.autoSize = TextFieldAutoSize.CENTER;
_rewardValue.align
```

```
= TextFormatAlign.CENTER;

if (type == REWARD_PALLADIUM)
{
_rewardSymbol = UIFactory.getBitmap('IconPalladiumBMD');
_bg.width = 38;
_rewardValue.width = 38;
}
else if (type == REWARD_EXP)
{
_reward = new Label(22, 0xf0f0f0, 140, 40);
_reward.autoSize = TextFieldAutoSize.LEFT;
_reward.align = TextFormatAlign.LEFT;
_reward.text = _exp;
} else if (type == REWARD_CREDITS)
_rewardSymbol = UIFactory.getBitmap('IconCreditBMD');
else if (type == BLUEPRINT)
{
_bg.width = 38;
_rewardValue.width = 38;
_reward = new Label(22, 0xf0f0f0, 140, 40);
_reward.autoSize = TextFieldAutoSize.LEFT;
_reward.align = TextFormatAlign.LEFT;
_reward.text = _blueprintPiece;
}

addChild(_bg);
addChild(_rewardValue);

if (_reward)
addChild(_reward);

if (_rewardSymbol)
addChild(_rewardSymbol);

layout();
}

public function layout():void
{
switch (_type)
{
case REWARD_CREDITS:
case REWARD_PALLADIUM:
_bg.x = _rewardSymbol.x + _rewardSymbol.width + 5;
_bg.y = 4;
break;
case REWARD_EXP:
case BLUEPRINT:
_reward.y = 2;
_bg.x
```

```
= _reward.x + _reward.textWidth + 4;
_bg.y = 4;
break;
}

_rewardValue.x = _bg.x + _bg.width * 0.5;
_rewardValue.y = _bg.y;

_rewardValue.text = StringUtil.commaFormatNumber(_value);
}

public function destroy():void
{
_bg = null;

_rewardSymbol = null;

if (_rewardValue)
_rewardValue.destroy();

_rewardValue = null;

if (_reward)
_reward.destroy();

_reward = null;
}

}
}

-----------------------------------
File 776: igw\com\ui\modal\achievements\AchievementView.as
package com.ui.modal.achievements
{
import com.enum.ui.LabelEnum;
import com.model.achievements.AchievementVO;
import com.model.asset.AssetVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.presenter.shared.IAchievementPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.accordian.AccordianComponent;
import com.ui.core.component.accordian.AccordianGroup;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import
```

```
flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

import org.shared.ObjectPool;

public class AchievementView extends View
{
private var _currentAchievements:Dictionary;
private var _currentScores:Dictionary;

private var _accordian:AccordianComponent;

private var _bg:DefaultWindowBG;

private var _totalAchievementsText:Label;

private var _achievementEntries:Dictionary;

private var _scrollbar:VScrollbar;

private var _maxHeight:int;
private var _completedAchievements:int;
private var _totalAchievements:int;

private var _scrollRect:Rectangle;

private var _holder:Sprite;

private var _achievementEntriesDictionary:Dictionary;

private var _titleText:String = 'CodeString.Achievements.Title'; //BADGES OF HONOR
private var _outOfString:String = 'CodeString.Shared.OutOf';
//[[Number.MinValue]]/[[Number.MaxValue]]

[PostConstruct]
override public function init():void
{
super.init();
_achievementEntries = new Dictionary;

_bg = ObjectPool.get(DefaultWindowBG);
var accordianOffset:int = 261;
//_bg.setBGSize(633, 530);
_bg.setBGSize(894-29, 535);
_bg.addTitle(_titleText, 240);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_accordian = ObjectPool.get(AccordianComponent);
_accordian.init(244,
```

```
52);
_accordian.x = _bg.bg.x + 14;
_accordian.y = _bg.bg.y + 5;
_accordian.addListener(onAccordianSelected);

_totalAchievementsText = UIFactory.getLabel(LabelEnum.H1, 100, 50);
_totalAchievementsText.align = TextFormatAlign.LEFT;
_totalAchievementsText.x = 242;
_totalAchievementsText.y = 4;

_holder = new Sprite();
_holder.x = accordianOffset + 12;//41;
_holder.y = _bg.bg.y + 5;

_maxHeight = 0;
_completedAchievements = 0;
_totalAchievements = 0;

_scrollRect = new Rectangle(_holder.x, _holder.y, 582+36, 514);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _bg.x + _bg.width - 25;
var scrollbarYPos:Number = _bg.y + 52;
_scrollbar.init(7, _scrollRect.height - 15, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 174;

addChild(_bg);
addChild(_holder);
addChild(_scrollbar);
addChild(_totalAchievementsText);
addChild(_accordian);

_achievementEntriesDictionary = new Dictionary;
var accordianGroups:Vector.<IPrototype> =
setUpAccordian(presenter.getFilterAchievementPrototypes());
setUpStars(presenter.getAchievementPrototypes());

if(accordianGroups.length > 0)
{
var firstGroup:String = accordianGroups[0].getValue('key');
setUp(presenter.getAchievementPrototypes(), firstGroup);
_accordian.setSelected(firstGroup, null);
}

presenter.onAddAchievementsUpdatedListener(onAchievementsUpdated);
```

```
presenter.requestAchievements();

addEffects();
effectsIN();
}

private function onAccordianSelected( groupID:String, subItemID:String, data:* ):void
{
setUp(presenter.getAchievementPrototypes(), groupID);
if(_currentAchievements && _currentScores)
{
_completedAchievements = 0;
onAchievementsUpdated(_currentAchievements, _currentScores);
}
}

private function setUpAccordian( v:Vector.<IPrototype> ):Vector.<IPrototype>
{
var len:uint = v.length;
var currentFilterAchievement:IPrototype;

v.sort(orderAccordianItems);

for (var i:uint = 0; i < len; ++i)
{
currentFilterAchievement = v[i];
_accordian.addGroup(currentFilterAchievement.getValue('key'),
currentFilterAchievement.getValue('uiName'));
}

return v;
}

private function orderAccordianItems( itemOne:IPrototype, itemTwo:IPrototype ):int
{
var sortOrderOne:Number = itemOne.getValue('sort');
var sortOrderTwo:Number = itemTwo.getValue('sort');

if (sortOrderOne < sortOrderTwo)
{
return -1;
} else if (sortOrderOne > sortOrderTwo)
{
return 1;
} else
{
return 0;
}
}
```

```
private function setUpStars( v:Vector.<IPrototype> ):void
{
var len:uint = v.length;
var currentEntry:AchievementDisplay;
var currentCategory:String;
var currentAchievement:IPrototype;
var currentAchievementAsset:AssetVO;
for (var i:uint = 0; i < len; ++i)
{
currentAchievement = v[i];
currentCategory = currentAchievement.getValue('category');
_achievementEntriesDictionary[currentCategory] = 0;
}
for (var i:uint = 0; i < len; ++i)
{
currentAchievement = v[i];
currentCategory = currentAchievement.getValue('category');
_achievementEntriesDictionary[currentCategory]++;
}
}

private function setUp( v:Vector.<IPrototype>, filterAchievement:String = ''):void
{
_achievementEntries = new Dictionary;
_holder.removeChildren();

var len:uint = v.length;
var currentEntry:AchievementDisplay;
var currentCategory:String;
var currentAchievement:IPrototype;
var currentAchievementAsset:AssetVO;
var currentFaction:String;
var currentFilterAchievement:String;
for (var i:uint = 0; i < len; ++i)
{
currentAchievement = v[i];
currentAchievementAsset = presenter.getAssetVOFromIPrototype(currentAchievement);
currentCategory = currentAchievement.getValue('category');
currentFaction = currentAchievement.getValue('factionType');

if(currentFaction && CurrentUser.faction != currentFaction)
{
continue;
}

currentFilterAchievement = currentAchievement.getValue('filterAchievement');//check na null

if(currentFilterAchievement == filterAchievement)
{
```

```
if (!(currentCategory in _achievementEntries))
{
currentEntry = new AchievementDisplay(currentCategory);
currentEntry.setUpStars(_achievementEntriesDictionary[currentCategory]);
currentEntry.onLoadImage.add(presenter.loadIcon);
currentEntry.onClaimReward.add(presenter.claimAchievementReward);
currentEntry.maxCredits = presenter.maxCredits;
currentEntry.getScore = presenter.getScoreValueByName;
_holder.addChild(currentEntry);
} else
currentEntry = _achievementEntries[currentCategory];

currentEntry.addAchievement(currentAchievement, currentAchievementAsset);

_achievementEntries[currentCategory] = currentEntry;
}
}

layout();
_scrollbar.resetScroll();
}

private function onAchievementsUpdated( achievements:Dictionary, scores:Dictionary ):void
{
_currentAchievements = achievements;
_currentScores = scores;

var currentEntry:AchievementDisplay;
var currentAchievementProgress:AchievementVO;
var achievementProtoName:String;
var achievementCategory:String;

for (var key:String in achievements)
{
currentAchievementProgress = achievements[key];
achievementCategory = currentAchievementProgress.category;
if (achievementCategory in _achievementEntries)
{
++_completedAchievements;
currentEntry = _achievementEntries[achievementCategory];
currentEntry.addAchievementProgress(currentAchievementProgress);
_achievementEntries[achievementCategory] = currentEntry;
}
}

for each (var entry:AchievementDisplay in _achievementEntries)
{
if (entry.scoreKey != null && entry.scoreKey != '')
{
if
```

```
(entry.scoreKey in scores)
entry.setScore(scores[entry.scoreKey].value);
else
entry.setScore(0);
}
}
_totalAchievementsText.setTextWithTokens(_outOfString,
{'[[Number.MinValue]]':_completedAchievements, '[[Number.MaxValue]]':_totalAchievements});

layout();
_scrollbar.resetScroll();
}

protected function layout():void
{
_totalAchievements = 0;
var entry:AchievementDisplay;
var holder:Vector.<AchievementDisplay> = new Vector.<AchievementDisplay>;
var yPos:int = 0;
var xPos:int = _holder.x;
var offset:Number = 174;
_maxHeight = 0;
for each (entry in _achievementEntries)
{
holder.push(entry);
++_totalAchievements;
}

holder.sort(orderByID);

for (var i:uint = 0; i < _totalAchievements; ++i)
{
entry = holder[i];
entry.x = xPos;
entry.y = yPos;
_maxHeight += offset;
yPos += offset;
}

_totalAchievements = 0;
holder.length = 0;
for each (entry in _achievementEntries)
{
holder.push(entry);
_totalAchievements += entry.achievementsCount();
}

_maxHeight -= 5;
_scrollbar.updateScrollableHeight(_maxHeight);
_totalAchievementsText.setTextWithTokens(_outOfString,
{'[[Number.MinValue]]':_completedAchievements,
```

```
'[[Number.MaxValue]]':_totalAchievements});
}

private function orderByID( achievementOne:AchievementDisplay,
achievementTwo:AchievementDisplay ):Number
{
if (!achievementOne)
return -1;
if (!achievementTwo)
return 1;

var achievementOneSort:Number = achievementOne.sort;
var achievementTwoSort:Number = achievementTwo.sort;

if(achievementOne._claimBtn.visible && !achievementTwo._claimBtn.visible)
return -1;
else if(!achievementOne._claimBtn.visible && achievementTwo._claimBtn.visible)
return 1;

if (achievementOneSort < achievementTwoSort)
return -1;
else if (achievementOneSort > achievementTwoSort)
return 1;

return 0;
}


private function onChangedScroll( percent:Number ):void
{
if (_scrollRect)
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;

if (_holder)
_holder.scrollRect = _scrollRect;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IAchievementPresenter ):void { _presenter = v; }
public function get presenter():IAchievementPresenter { return
IAchievementPresenter(_presenter); }

override public function destroy():void
{
presenter.onRemoveAchievementsUpdatedListener(onAchievementsUpdated);
super.destroy();

ObjectPool.give(_accordian);
```

```
_accordian = null;

_currentAchievements = null;
_currentScores = null;

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_totalAchievementsText)
_totalAchievementsText.destroy();

_totalAchievementsText = null;

for each (var entry:AchievementDisplay in _achievementEntries)
{
entry.destroy();
entry = null;
}
_achievementEntries = null;

_scrollbar = null;
_maxHeight = 0;

_scrollRect = null;

_holder = null;
}
}
}

-----------------------------------
File 777: igw\com\ui\modal\alliances\alliance\AllianceEditInfoView.as
package com.ui.modal.alliances.alliance
{
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.ui.core.ScaleBitmap;

import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

public
```

```
class AllianceEditInfoView extends View
{
private var _acceptChangesBtn:BitmapButton;
private var _bg:ScaleBitmap;
private var _bodyText:Label;
private var _bodyTextBG:ScaleBitmap;
private var _bodyTextHolder:Sprite;
private var _callback:Function;
private var _closeBtn:BitmapButton;
private var _defaultBodyText:String;
private var _maxChars:uint;
private var _scrollbar:VScrollbar;
private var _scrollRect:Rectangle;
private var _title:Label;
private var _titleText:String;

private var _acceptText:String = 'CodeString.Shared.Accept'; //ACCEPT
private var _enterText:String = 'CodeString.Shared.EnterText'; //Enter Text...

[PostConstruct]
public override function init():void
{
super.init();

_bg = PanelFactory.getScaleBitmapPanel('WindowContextMenuBMD', 549, 300, new
Rectangle(190, 120, 5, 5))
addChild(_bg);

_bodyTextBG = PanelFactory.getScaleBitmapPanel('AllianceTextboxBMD', 500, 107, new
Rectangle(15, 11, 2, 2));
_bodyTextBG.x = 22;
_bodyTextBG.y = 85;

_title = new Label(22, 0xf0f0f0, 150, 30);
_title.allCaps = true;
_title.align = TextFormatAlign.LEFT;
_title.x = 25;
_title.y = 12;
_title.text = _titleText;

_bodyText = new Label(18, 0xa9dcff, 450, 235, true);
_bodyText.align = TextFormatAlign.LEFT;
_bodyText.maxChars = _maxChars;
_bodyText.multiline = true;
_bodyText.allowInput = true;
_bodyText.clearOnFocusIn = true;
_bodyText.letterSpacing = .8;
_bodyText.addLabelColor(0xbdfefd, 0x000000);
_bodyText.updateInputText(_enterText);
if (_defaultBodyText != '')
_bodyText.text
```

```
= _defaultBodyText;
_bodyText.addEventListener(Event.CHANGE, onTextUpdated, false, 0, true);

_bodyTextHolder = new Sprite();
_bodyTextHolder.x = _bodyTextBG.x + 5;
_bodyTextHolder.y = _bodyTextBG.y + 3;
_bodyTextHolder.addChild(_bodyText);

_scrollRect = new Rectangle(0, 0, _bodyTextHolder.width, _bodyTextBG.height - 5);
_scrollRect.y = 0;
_bodyTextHolder.scrollRect = _scrollRect;

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _bodyTextBG.x + _bodyTextBG.width - 1;
var scrollbarYPos:Number = _bodyTextBG.y;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this, _bodyTextHolder);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_bodyText.textHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 7;

_acceptChangesBtn = ButtonFactory.getBitmapButton('BlueBtnCNeutralBMD', 0, 0,
_acceptText, 0xf0f0f0, 'BlueBtnCRollOverBMD', 'BlueBtnCSelectedBMD', null,
'BlueBtnCSelectedBMD');
_acceptChangesBtn.x = _bodyTextBG.x + (_bodyTextBG.width - _acceptChangesBtn.width) *
0.5;
_acceptChangesBtn.y = 240;
_acceptChangesBtn.addEventListener(MouseEvent.CLICK, onAcceptChanges, false, 0, true);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 13);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

addChild(_bg);
addChild(_closeBtn);
addChild(_bodyTextBG);
addChild(_title);
addChild(_bodyTextHolder);
addChild(_scrollbar);
addChild(_acceptChangesBtn)

addEffects();
effectsIN();
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_bodyText.textHeight - _scrollRect.height) * percent;
_bodyTextHolder.scrollRect = _scrollRect;
}
```

```
private function onTextUpdated( e:Event ):void
{
_scrollbar.updateScrollableHeight(_bodyText.textHeight);

var car:int = _bodyText.caretIndex;
var rect:Rectangle = _bodyText.getCharBoundaries(car - 1);

if (rect != null)
{
if (rect.y + rect.height > _scrollRect.height)
{
var percent:Number = ((rect.y + rect.height) / _bodyText.textHeight)
_scrollbar.updateScrollPercent(percent);
} else if (rect.y + rect.height < _scrollRect.height && _scrollbar.percent != 0)
_scrollbar.updateScrollPercent(0);
}
}

private function onAcceptChanges( e:MouseEvent ):void
{
var newText:String = _bodyText.text;
if ((_defaultBodyText == '' || _defaultBodyText != '' && newText != _defaultBodyText) &&
_callback != null)
{
_callback(newText);
destroy();
}
}

public function set bodyText( v:String ):void { _defaultBodyText = v; }
public function set callback( v:Function ):void { _callback = v; }
public function set maxChars( v:uint ):void { _maxChars = v; }
public function set titleText( v:String ):void { _titleText = v; }

override public function destroy():void
{
super.destroy();
}
}
}

-----------------------------------
File 778: igw\com\ui\modal\alliances\alliance\AllianceInfoDisplay.as
package com.ui.modal.alliances.alliance
{
import com.enum.server.AllianceRankEnum;
import com.enum.server.AllianceResponseEnum;
import com.enum.ui.ButtonEnum;
import
```

```
com.enum.ui.LabelEnum;
import com.model.alliance.AllianceVO;
import com.model.player.CurrentUser;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

public class AllianceInfoDisplay extends Sprite
{
public var onMotdUpdated:Function;
public var onDescriptionUpdated:Function;
public var onPublicChanged:Function;
public var onLeaveAlliance:Function;
public var onSendAllianceMail:Function;

private var _membersCount:Label;
private var _motdTitle:Label;
private var _motd:Label;
private var _description:Label;
private var _descriptionTitle:Label;
private var _rank:Label;
private var _checkboxText:Label;

private var _motdBG:Bitmap;
private var _descriptionBG:ScaleBitmap;

private var _motdHolder:Sprite;
private var _descriptionHolder:Sprite;

private var _leaveBtn:BitmapButton;
private var _publicCheckbox:BitmapButton;
private var _editMOTD:BitmapButton;
private var _editDescription:BitmapButton;
private var _messageAllBtn:BitmapButton;

private
```

```
var _motdScrollbar:VScrollbar;
private var _descriptionScrollbar:VScrollbar;

private var _motdScrollRect:Rectangle;
private var _descriptionScrollRect:Rectangle;

private var _alliance:AllianceVO;

private var _tooltips:Tooltips;

private var _messageOfTheDayText:String = 'CodeString.AllianceInfoDisplay.MotdTitle';
//MESSAGE OF THE DAY
private var _descriptionText:String = 'CodeString.AllianceInfoDisplay.DescriptionTitle';
//DESCRIPTION
private var _leaveAllianceText:String = 'CodeString.AllianceInfoDisplay.LeaveAlliance'; //QUIT
private var _publicAllianceText:String = 'CodeString.AllianceInfoDisplay.PublicAlliance'; //Public
Alliance
private var _publicAllianceDescriptionText:String = 'CodeString.Alliance.PublicDescription'
//Public alliances can be joined by anyone in your faction!
private var _alliancePlayerRankText:String = 'CodeString.AllianceInfoDisplay.PlayerRank' //Your
Rank: [[String:CurrentPlayerRank]]
private var _allianceMemberCountText:String = 'CodeString.AllianceInfoDisplay.MemberCount'
//Members: [[Number:CurrentMemberCount]]/[[Number:MaxMemberCount]]
private var _messageAllText:String = 'CodeString.AllianceInfoDisplay.MessageAll';

public function AllianceInfoDisplay()
{
super();

_motdBG = PanelFactory.getScaleBitmapPanel('AllianceTextboxBMD', 500, 57, new
Rectangle(15, 11, 2, 2));
_motdBG.x = 6;
_motdBG.y = 38;

_descriptionBG = PanelFactory.getScaleBitmapPanel('AllianceTextboxBMD', 500, 107, new
Rectangle(15, 11, 2, 2));
_descriptionBG.x = 6;
_descriptionBG.y = 156;

_membersCount = new Label(20, 0xf0f0f0, 150, 25);
_membersCount.align = TextFormatAlign.LEFT;
_membersCount.x = 385;
_membersCount.y = 133;

_motdTitle = new Label(18, 0xa9dcff, 150, 25);
_motdTitle.align = TextFormatAlign.LEFT;
_motdTitle.x = 10;
_motdTitle.y = 15;
_motdTitle.text = _messageOfTheDayText;

_descriptionTitle
```

```
= new Label(18, 0xa9dcff, 150, 25);
_descriptionTitle.align = TextFormatAlign.LEFT;
_descriptionTitle.x = 10;
_descriptionTitle.y = 133;
_descriptionTitle.text = _descriptionText;

_rank = new Label(18, 0xf0f0f0, 150, 25, false);
_rank.align = TextFormatAlign.LEFT;
_rank.x = 10;
_rank.y = 270;

_motd = new Label(18, 0xa9dcff, 450);
_motd.constrictTextToSize = false;
_motd.autoSize = TextFieldAutoSize.LEFT;
_motd.align = TextFormatAlign.LEFT;
_motd.multiline = true;

_motdHolder = new Sprite();
_motdHolder.x = _motdBG.x + 5;
_motdHolder.y = _motdBG.y + 3;
_motdHolder.addChild(_motd);

_description = new Label(18, 0xa9dcff, 450);
_description.constrictTextToSize = false;
_description.autoSize = TextFieldAutoSize.LEFT;
_description.align = TextFormatAlign.LEFT;
_description.multiline = true;

_descriptionHolder = new Sprite();
_descriptionHolder.x = _descriptionBG.x + 5;
_descriptionHolder.y = _descriptionBG.y + 3;
_descriptionHolder.addChild(_description);

_leaveBtn = ButtonFactory.getBitmapButton('CancelBtnNeutralBMD', 0, 0, _leaveAllianceText,
0xF58993, 'CancelBtnRollOverBMD', 'CancelBtnDownBMD');
_leaveBtn.x = 372;
_leaveBtn.y = _rank.y;
_leaveBtn.addEventListener(MouseEvent.CLICK, onLeaveAllianceClick, false, 0, true);

_editMOTD = ButtonFactory.getBitmapButton('EditBtnUpBMD', 0, 0, '', 0, 'EditBtnRollOverBMD',
'EditBtnDownBMD', null, 'EditBtnDownBMD');
_editMOTD.x = _motdBG.x + _motdBG.width - _editMOTD.width;
_editMOTD.y = _motdBG.y - _editMOTD.height * 0.5;
_editMOTD.addEventListener(MouseEvent.CLICK, onMessageOfTheDayClick, false, 0, true);

_editDescription = ButtonFactory.getBitmapButton('EditBtnUpBMD', 0, 0, '', 0,
'EditBtnRollOverBMD', 'EditBtnDownBMD', null, 'EditBtnDownBMD');
_editDescription.x = _descriptionBG.x + _descriptionBG.width - _editDescription.width;
_editDescription.y = _descriptionBG.y - _editDescription.height * 0.5;
_editDescription.addEventListener(MouseEvent.CLICK, onDescriptionClick, false, 0, true);

_publicCheckbox
```

```
= ButtonFactory.getBitmapButton('CheckboxBtnUncheckedBMD', 0, 0, '', 0, null,
'CheckboxBtnUncheckedBMD', null, 'CheckboxBtnCheckedBMD');
_publicCheckbox.x = _rank.x;
_publicCheckbox.y = _rank.y + 30;
_publicCheckbox.addEventListener(MouseEvent.CLICK, onIsPublicClick, false, 0, true);
_publicCheckbox.selectable = true;

_messageAllBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 144, 29, 362, 334,
_messageAllText, LabelEnum.H1);
_messageAllBtn.addEventListener(MouseEvent.CLICK, onSendAllianceMailClick, false, 0, true);

_checkboxText = new Label(18, 0xa9dcff, 469, 30, true);
_checkboxText.align = TextFormatAlign.LEFT;
_checkboxText.text = _publicAllianceText;
_checkboxText.x = _publicCheckbox.x + _publicCheckbox.width + 5;
_checkboxText.y = _publicCheckbox.y + (_checkboxText.textHeight - _publicCheckbox.height) *
0.5;
_checkboxText.visible = false;

_motdScrollRect = new Rectangle(0, 0, _motdHolder.width, _motdBG.height - 5);
_motdScrollRect.y = 0;
_motdHolder.scrollRect = _motdScrollRect;

_descriptionScrollRect = new Rectangle(0, 0, _descriptionHolder.width, _descriptionBG.height -
5);
_descriptionScrollRect.y = 0;
_descriptionHolder.scrollRect = _descriptionScrollRect;

_descriptionScrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _descriptionBG.x + _descriptionBG.width - 3;
var scrollbarYPos:Number = _descriptionBG.y;
_descriptionScrollbar.init(7, _descriptionScrollRect.height, scrollbarXPos, scrollbarYPos,
dragBarBGRect, '', 'ScrollbarBMD', '', false, this, _descriptionHolder);
_descriptionScrollbar.onScrollSignal.add(onChangedDescriptionScroll);
_descriptionScrollbar.updateScrollableHeight(_description.textHeight);
_descriptionScrollbar.updateDisplayedHeight(_descriptionScrollRect.height);
_descriptionScrollbar.maxScroll = 7;

_motdScrollbar = new VScrollbar();
scrollbarXPos = _motdBG.x + _motdBG.width - 3;
scrollbarYPos = _motdBG.y;
_motdScrollbar.init(7, _motdScrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this, _motdHolder);
_motdScrollbar.onScrollSignal.add(onChangedMOTDScroll);
_motdScrollbar.updateScrollableHeight(_motd.textHeight);
_motdScrollbar.updateDisplayedHeight(_motdScrollRect.height);
_motdScrollbar.maxScroll = 7;

addChild(_motdBG);
addChild(_descriptionBG);
```

```
addChild(_membersCount);
addChild(_motdTitle);
addChild(_descriptionTitle);
addChild(_rank);
addChild(_motdHolder);
addChild(_motdScrollbar);
addChild(_descriptionHolder);
addChild(_descriptionScrollbar);
addChild(_publicCheckbox);
addChild(_checkboxText);
addChild(_editMOTD);
addChild(_editDescription);
addChild(_leaveBtn);
addChild(_messageAllBtn);

if (_alliance)
onAllianceUpdated(_alliance);
}

public function handleAllianceMessage( messageEnum:int ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_NAMEINUSE:
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_UNKNOWN:
break;
}
}

public function onAllianceUpdated( v:AllianceVO ):void
{
_alliance = v;

if (CurrentUser.alliance == v.key)
{
if (_rank)
_rank.setTextWithTokens(_alliancePlayerRankText,
{'[[String:CurrentPlayerRank]]':CommonFunctionUtil.getAllianceRankName(CurrentUser.allianceRank)});

if (CurrentUser.allianceRank > AllianceRankEnum.MEMBER)
{
_messageAllBtn.visible = _editMOTD.visible = _editDescription.visible = true;

} else
{
_messageAllBtn.visible = _editMOTD.visible = _editDescription.visible = false;
}

if (CurrentUser.allianceRank == AllianceRankEnum.LEADER)
{
```

```
_publicCheckbox.visible = _checkboxText.visible = true;
_leaveBtn.visible = (v.memberCount <= 1) ? true : false;
} else
_publicCheckbox.visible = _checkboxText.visible = false;

} else
{
_rank.text = '';
_leaveBtn.visible = _publicCheckbox.visible = _checkboxText.visible = _messageAllBtn.visible =
_editMOTD.visible = _editDescription.visible = false;
}

_motd.text = v.motd;
_description.text = v.description;
_publicCheckbox.selected = v.isPublic;
_membersCount.setTextWithTokens(_allianceMemberCountText,
{'[[Number:CurrentMemberCount]]':v.memberCount, '[[Number:MaxMemberCount]]':1000});

_motdScrollbar.updateScrollableHeight(_motd.textHeight);
_descriptionScrollbar.updateScrollableHeight(_description.textHeight);
}

private function onSendAllianceMailClick( e:MouseEvent ):void
{
if (onSendAllianceMail != null)
onSendAllianceMail();
}

private function onIsPublicClick( e:MouseEvent ):void
{
if (onPublicChanged != null)
onPublicChanged(_publicCheckbox.selected);
}

private function onMessageOfTheDayClick( e:MouseEvent ):void
{
if (onMotdUpdated != null)
onMotdUpdated(_motd.text);
}

private function onDescriptionClick( e:MouseEvent ):void
{
if (onDescriptionUpdated != null)
onDescriptionUpdated(_description.text);
}

private function onLeaveAllianceClick( e:MouseEvent ):void
{
if (onLeaveAlliance != null)
onLeaveAlliance();
```

```
}

private function onChangedMOTDScroll( percent:Number ):void
{
_motdScrollRect.y = (_motd.textHeight - _motdScrollRect.height) * percent;
_motdHolder.scrollRect = _motdScrollRect;
}

private function onChangedDescriptionScroll( percent:Number ):void
{
_descriptionScrollRect.y = (_description.textHeight - _descriptionScrollRect.height) * percent;
_descriptionHolder.scrollRect = _descriptionScrollRect;
}

public function setEnabled( state:Boolean ):void
{
_leaveBtn.enabled = state;
_editMOTD.enabled = state;
_editDescription.enabled = state;
_publicCheckbox.enabled = state;
_messageAllBtn.enabled = state;
}

public function setVisible( state:Boolean ):void
{
_leaveBtn.visible = state;
_editMOTD.visible = state;
_editDescription.visible = state;
_publicCheckbox.visible = state;
_messageAllBtn.visible = state;
}

[Inject]
public function set tooltips( value:Tooltips ):void
{
_tooltips = value;

if (_publicCheckbox)
_tooltips.addTooltip(_publicCheckbox, this, null,
Localization.instance.getString(_publicAllianceDescriptionText));
}

public function destroy():void
{
if (_tooltips)
_tooltips.removeTooltip(null, this);

_descriptionScrollRect = null;
_alliance = null;
onMotdUpdated
```

```
= null;
onDescriptionUpdated = null;
onPublicChanged = null;
onLeaveAlliance = null;

if (_membersCount)
_membersCount.destroy();

_membersCount = null;

if (_motdTitle)
_motdTitle.destroy();

_motdTitle = null;

if (_motd)
_motd.destroy();

_motd = null;

if (_description)
_description.destroy();

_description = null;

if (_descriptionTitle)
_descriptionTitle.destroy();

_descriptionTitle = null;

if (_rank)
_rank.destroy();

_rank = null;

if (_checkboxText)
_checkboxText.destroy();

_checkboxText = null;

_motdBG = null;
_descriptionBG = null;
_descriptionHolder = null;

if (_leaveBtn)
{
_leaveBtn.removeEventListener(MouseEvent.CLICK, onLeaveAllianceClick);
_leaveBtn.destroy();
}

_leaveBtn
```

```
= null;

if (_publicCheckbox)
{
_publicCheckbox.removeEventListener(MouseEvent.CLICK, onIsPublicClick);
_publicCheckbox.destroy();
}

_publicCheckbox = null;

if (_editMOTD)
{
_editMOTD.removeEventListener(MouseEvent.CLICK, onMessageOfTheDayClick);
_editMOTD.destroy();
}

_editMOTD = null;

if (_editDescription)
{
_editDescription.removeEventListener(MouseEvent.CLICK, onDescriptionClick);
_editDescription.destroy();
}

_editDescription = null;

if (_descriptionScrollbar)
_descriptionScrollbar.destroy();

_descriptionScrollbar = null;

if (_messageAllBtn)
{
_messageAllBtn.removeEventListener(MouseEvent.CLICK, onSendAllianceMail);
_messageAllBtn.destroy();
}
_messageAllBtn = null;
}
}
}

-----------------------------------
File 779: igw\com\ui\modal\alliances\alliance\AllianceMemberDisplay.as
package com.ui.modal.alliances.alliance
{
import com.Application;
import com.controller.keyboard.KeyboardController;
import com.controller.keyboard.KeyboardKey;
import com.enum.server.AllianceResponseEnum;
import com.model.alliance.AllianceMemberVO;
import
```

```
com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.display.Stage;
import flash.events.Event;
import flash.events.FocusEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;

public class AllianceMemberDisplay extends Sprite
{
private var _name:Label;
private var _rank:Label;
private var _level:Label;
private var _actions:Label;
private var _search:Label;

private var _alliancesBG:Bitmap;
private var _searchBG:Bitmap;

private var _allianceHolder:Sprite;

private var _searchBtn:BitmapButton;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _members:Vector.<AllianceMemberEntry>;
private var _visibleMembers:Vector.<AllianceMemberEntry>;

private var _allianceKey:String;

private var _stage:Stage;
private var _keyboard:KeyboardController;

public var onPromoteMember:Function;
public var onDemoteMember:Function;
public var onRemoveMember:Function;
public var onShowProfile:Function;

private
```

```
var _nameText:String = 'CodeString.AllianceMemberDisplay.Name'; //NAME
private var _rankText:String = 'CodeString.AllianceMemberDisplay.Rank'; //RANK
private var _levelText:String = 'CodeString.AllianceMemberDisplay.Level'; //LEVEL
private var _actionText:String = 'CodeString.AllianceMemberDisplay.Actions'; //ACTIONS
private var _searchText:String = 'CodeString.AllianceMemberDisplay.Search'; //Search
Members....

public function AllianceMemberDisplay()
{
super();

_stage = Application.STAGE;

_visibleMembers = new Vector.<AllianceMemberEntry>;
_members = new Vector.<AllianceMemberEntry>;

_searchBG = PanelFactory.getPanel('AllianceMemberSearchBMD');
_searchBG.x = 273;
_searchBG.y = 24;

_name = new Label(18, 0xfbefaf, 150, 25);
_name.align = TextFormatAlign.LEFT;
_name.x = 51;
_name.y = 68;
_name.text = _nameText;

_rank = new Label(18, 0xfbefaf, 150, 25);
_rank.align = TextFormatAlign.LEFT;
_rank.x = 217;
_rank.y = 68;
_rank.text = _rankText;

_level = new Label(18, 0xfbefaf, 150, 25);
_level.align = TextFormatAlign.LEFT;
_level.x = 337;
_level.y = 68;
_level.text = _levelText;

_actions = new Label(18, 0xfbefaf, 150, 25);
_actions.align = TextFormatAlign.LEFT;
_actions.x = 391;
_actions.y = 68;
_actions.text = _actionText;

_search = new Label(18, 0xf0f0f0, 184, 30);
_search.align = TextFormatAlign.LEFT;
_search.text = _searchText;
_search.x = _searchBG.x + 8;
_search.y = _searchBG.y;
_search.maxChars = 20;
_search.allowInput
```

```
= true;
_search.clearOnFocusIn = true;
_search.letterSpacing = .8;
_search.addLabelColor(0xbdfefd, 0x000000);
_search.addEventListener(Event.CHANGE, onChanged, false, 0, true);

_searchBtn = ButtonFactory.getBitmapButton('AllianceSearchBtnUpBMD', _searchBG.x + 194,
_searchBG.y + 3, '', 0, 'AllianceSearchBtnRollOverBMD');

_alliancesBG = PanelFactory.getPanel('AllianceMemberBGBMD');
_alliancesBG.x = 36;
_alliancesBG.y = 95;

_allianceHolder = new Sprite();
_allianceHolder.x = _alliancesBG.x;
_allianceHolder.y = _alliancesBG.y;
_maxHeight = 0;

_scrollRect = new Rectangle(_allianceHolder.x, _allianceHolder.y, 460, 287);
_scrollRect.y = 0;
_allianceHolder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _alliancesBG.x + _alliancesBG.width + 5;
var scrollbarYPos:Number = _alliancesBG.y;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this, _allianceHolder);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 23;

addChild(_name);
addChild(_rank);
addChild(_level);
addChild(_actions);
addChild(_alliancesBG);
addChild(_allianceHolder);
addChild(_searchBG);
addChild(_search);
addChild(_searchBtn);
addChild(_scrollbar);
}

public function handleAllianceMessage( messageEnum:int ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.JOIN_FAILED_TOOMANYPLAYERS:
break;
```

```
}
}

private function filterMember( filterBy:String ):void
{
if (_allianceHolder.numChildren > 0)
_allianceHolder.removeChildren(0, (_allianceHolder.numChildren - 1));

_visibleMembers.length = 0;

var len:uint = _members.length;
var currentEntry:AllianceMemberEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _members[i];
if (filterBy == '' || filterBy == _search.inputMessage.toLowerCase() ||
currentEntry.filterBy.indexOf(filterBy) != -1)
{
_allianceHolder.addChild(currentEntry);
_visibleMembers.push(currentEntry);
}
}
_visibleMembers.sort(memberSortingFunction);
layout();
_scrollbar.resetScroll();
}

private function memberSortingFunction(entryA:AllianceMemberEntry,
entryB:AllianceMemberEntry):Number
{
if (entryA.rank < entryB.rank) return 1;
if (entryA.rank > entryB.rank) return -1;

//same rank so let's check xp (implicitly level)
if (entryA.xp > entryB.xp) return -1;
if (entryA.xp < entryB.xp) return 1;

return 0;
}

public function onMembersUpdated( v:Vector.<AllianceMemberVO> ):void
{
var len:uint = v.length;
var currentEntryCount:uint = _members.length;
var currentEntry:AllianceMemberEntry;
var i:uint = 0;
for (; i < len; ++i)
{
if (i < currentEntryCount)
{
```

```
currentEntry = _members[i];
currentEntry.update(v[i])
} else
{
currentEntry = new AllianceMemberEntry(i + 1, v[i], _allianceKey);
currentEntry.onShowProfile.add(onShowProfile);
currentEntry.onPromoteMember.add(onPromoteMember);
currentEntry.onDemoteMember.add(onDemoteMember);
currentEntry.onRemoveMember.add(onRemoveMember);
_members.push(currentEntry);
}
_allianceHolder.addChild(currentEntry);

}

if (i < currentEntryCount)
{
var startIndex:uint = i;
for (; i < currentEntryCount; ++i)
{
currentEntry = _members[i];
currentEntry.destroy();
currentEntry = null;
}
_members.splice(startIndex, i - startIndex);
}

onChanged();
layout();
}

private function layout():void
{
var len:uint = _visibleMembers.length;
var selection:AllianceMemberEntry;
var yPos:int = 0;
var xPos:int = _allianceHolder.x;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _visibleMembers[i];
selection.x = xPos;
selection.y = yPos;
_maxHeight += selection.height - 1;
yPos += selection.height - 1;
}
_scrollbar.updateScrollableHeight(_maxHeight);
}

private
```

```
function onChanged( e:Event = null ):void
{
filterMember(_search.text.toLowerCase());
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_allianceHolder.scrollRect = _scrollRect;
}

[Inject]
public function set keyboard( value:KeyboardController ):void
{
_keyboard = value;
_keyboard.addKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);
}

private function onEnterPress( keyCode:uint ):void
{
if (_stage.focus == _search)
{
filterMember(_search.text.toLowerCase());
addEventListener(Event.ENTER_FRAME, removeFocus, false, 0, true);
}
}

private function removeFocus( e:Event ):void
{
removeEventListener(Event.ENTER_FRAME, removeFocus);
if (_stage)
_stage.focus = Application.STAGE;
}

public function set allianceKey( v:String ):void
{
_allianceKey = v;
}

public function destroy():void
{
_alliancesBG = null;
_searchBG = null;
_allianceHolder = null;
_scrollRect = null;

onPromoteMember = null;
onDemoteMember = null;
onRemoveMember = null;

if
```

```
(_name)
_name.destroy();

_name = null;

if (_rank)
_rank.destroy();

_rank = null;

if (_level)
_level.destroy();

_level = null;

if (_actions)
_actions.destroy();

_actions = null;

if (_search)
_search.destroy();

_search = null;

if (_searchBtn)
_searchBtn.destroy();

_searchBtn = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar = null;

_visibleMembers.length = 0;
var len:uint = _members.length;
var currentEntry:AllianceMemberEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _members[i];
currentEntry.destroy();
currentEntry = null;
}
_members.length = 0;

_allianceKey = '';

_stage = null;

if
```

```
(_keyboard)
_keyboard.removeKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);

_keyboard = null;
}
}
}

-----------------------------------
File 780: igw\com\ui\modal\alliances\alliance\AllianceMemberEntry.as
package com.ui.modal.alliances.alliance
{
import com.enum.server.AllianceRankEnum;
import com.model.alliance.AllianceMemberVO;
import com.model.player.CurrentUser;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;

public class AllianceMemberEntry extends Sprite
{
private var _bg:Bitmap;

private var _name:Label;
private var _rank:Label;
private var _level:Label;

private var _promoteBtn:BitmapButton;
private var _demoteBtn:BitmapButton;
private var _removeBtn:BitmapButton;

private var _member:AllianceMemberVO;
private var _allianceKey:String;

public var onPromoteMember:Signal;
public var onDemoteMember:Signal;
public var onRemoveMember:Signal;
public var onShowProfile:Signal;

private var WIDTH:Number = 460;
private
```

```
var HEIGHT:Number = 25;

public function AllianceMemberEntry( index:uint, member:AllianceMemberVO,
allianceKey:String )
{
buttonMode = true;

_allianceKey = allianceKey;

onPromoteMember = new Signal(AllianceMemberVO);
onDemoteMember = new Signal(AllianceMemberVO);
onRemoveMember = new Signal(AllianceMemberVO);
onShowProfile = new Signal(AllianceMemberVO);

if ((index + 1) % 2 == 0)
{
_bg = PanelFactory.getScaleBitmapPanel('AllianceMemberRowBMD', WIDTH, HEIGHT, new
Rectangle(15, 11, 2, 2));
}

var center:Number = HEIGHT * 0.5 - 9;

_name = new Label(16, 0xf0f0f0, 213, 39, false);
_name.constrictTextToSize = false;
_name.align = TextFormatAlign.LEFT;

_rank = new Label(16, 0xf0f0f0, 110, 39);
_rank.constrictTextToSize = false;
_rank.align = TextFormatAlign.LEFT;


_level = new Label(16, 0xf0f0f0, 22, 39, false);
_level.constrictTextToSize = false;
_level.align = TextFormatAlign.CENTER;

_promoteBtn = ButtonFactory.getBitmapButton('AlliancePromoteUpBMD', 0, 0, '', 0,
'AlliancePromoteRollOverBMD');
_promoteBtn.addEventListener(MouseEvent.CLICK, onPromoteMemberClick, false, 0, true);

_demoteBtn = ButtonFactory.getBitmapButton('AllianceDemoteUpBtnBMD', 0, 0, '', 0,
'AllianceDemoteRollOverBtnBMD');
_demoteBtn.addEventListener(MouseEvent.CLICK, onDemoteMemberClick, false, 0, true);

_removeBtn = ButtonFactory.getBitmapButton('AllianceRemoveUpBMD', 0, 0, '', 0,
'AllianceRemoveRollOverBMD');
_removeBtn.addEventListener(MouseEvent.CLICK, onRemoveMemberClick, false, 0, true);

addEventListener(MouseEvent.CLICK, onMemberClick, false, 0, true);

if ((index + 1) % 2 == 0)
addChild(_bg);
```

```
addChild(_rank);
addChild(_name);
addChild(_level);
addChild(_promoteBtn);
addChild(_demoteBtn);
addChild(_removeBtn);

update(member);
}

public function update( member:AllianceMemberVO ):void
{
_member = member;

_name.text = _member.name;
_level.text = String(CommonFunctionUtil.findPlayerLevel(_member.xp));
_rank.text = CommonFunctionUtil.getAllianceRankName(_member.rank);

if (CurrentUser.alliance == _allianceKey && CurrentUser.allianceRank >
AllianceRankEnum.MEMBER && member.key != CurrentUser.id)
{
if (member.rank < CurrentUser.allianceRank && (CurrentUser.allianceRank ==
AllianceRankEnum.LEADER || CurrentUser.allianceRank == AllianceRankEnum.OFFICER))
{
if (CurrentUser.allianceRank == AllianceRankEnum.LEADER && member.rank <
AllianceRankEnum.LEADER)
_promoteBtn.visible = true;
else if (CurrentUser.allianceRank == AllianceRankEnum.OFFICER && member.rank <
AllianceRankEnum.MEMBER)
_promoteBtn.visible = true;
else
_promoteBtn.visible = false;

if (member.rank > AllianceRankEnum.RECRUIT)
_demoteBtn.visible = true;
else
_demoteBtn.visible = false;
} else
{
_promoteBtn.visible = false;
_demoteBtn.visible = false;
}

if (CurrentUser.allianceRank > member.rank)
_removeBtn.visible = true;
else
_removeBtn.visible = false;

}
```

```
else
{
_promoteBtn.visible = false;
_demoteBtn.visible = false;
_removeBtn.visible = false;
}

layout();
}

private function layout():void
{
var center:Number = HEIGHT * 0.5 - 9;

_name.x = 7;
_name.y = center;

_rank.x = 177;
_rank.y = center;

_level.x = 306;
_level.y = center;

var btnXPos:Number = 358;
if (_promoteBtn.visible)
{
_promoteBtn.x = btnXPos;
_promoteBtn.y = 3;
btnXPos = 394;
}

if (_demoteBtn.visible)
{
_demoteBtn.x = btnXPos;
_demoteBtn.y = 3;
btnXPos = 430;
}

if (_removeBtn.visible)
{
_removeBtn.x = btnXPos;
_removeBtn.y = 5;
}

}

private function onMemberClick( e:MouseEvent ):void
{
onShowProfile.dispatch(_member);
}

private
```

```
function onPromoteMemberClick( e:MouseEvent ):void
{
e.stopImmediatePropagation();
onPromoteMember.dispatch(_member);
}

private function onDemoteMemberClick( e:MouseEvent ):void
{
e.stopImmediatePropagation();
onDemoteMember.dispatch(_member);
}

private function onRemoveMemberClick( e:MouseEvent ):void
{
e.stopImmediatePropagation();
onRemoveMember.dispatch(_member);
}

public function get filterBy():String { return _member.name.toLowerCase(); }
public function get xp():int { return _member.xp; }
public function get rank():int { return _member.rank; }

override public function get height():Number { return HEIGHT; }
override public function get width():Number { return WIDTH; }

public function destroy():void
{

removeEventListener(MouseEvent.CLICK, onMemberClick);

if (onPromoteMember)
onPromoteMember.removeAll();

onPromoteMember = null;

if (onDemoteMember)
onDemoteMember.removeAll();

onDemoteMember = null;

if (onRemoveMember)
onRemoveMember.removeAll();

onRemoveMember = null;

_bg = null;

if (_rank)
_rank.destroy();
_rank = null;

if
```

```
(_name)
_name.destroy();
_name = null;

if (_level)
_level.destroy();
_level = null;

if (_promoteBtn)
{
_promoteBtn.removeEventListener(MouseEvent.CLICK, onPromoteMemberClick);
_promoteBtn.destroy();
}
_promoteBtn = null;

if (_demoteBtn)
{
_demoteBtn.removeEventListener(MouseEvent.CLICK, onDemoteMemberClick);
_demoteBtn.destroy();
}
_demoteBtn = null;

if (_removeBtn)
{
_removeBtn.removeEventListener(MouseEvent.CLICK, onRemoveMemberClick);
_removeBtn.destroy();
}
_removeBtn = null;
}
}
}

-----------------------------------
File 781: igw\com\ui\modal\alliances\alliance\AllianceView.as
package com.ui.modal.alliances.alliance
{
import com.enum.server.AllianceRankEnum;
import com.enum.server.AllianceResponseEnum;
import com.enum.ui.ButtonEnum;
import com.model.alliance.AllianceMemberVO;
import com.model.alliance.AllianceVO;
import com.model.player.CurrentUser;
import com.presenter.shared.IAlliancePresenter;
import com.ui.core.ButtonPrototype;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.hud.shared.mail.NewMailView;
import com.ui.modal.ButtonFactory;
import com.ui.modal.playerinfo.PlayerProfileView;

import
```

```
flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

public class AllianceView extends View
{
private var _bg:Bitmap;
private var _closeBtn:BitmapButton;

private var _currentState:Sprite;

private var _title:Label;

private var _infoBtn:BitmapButton;
private var _membersBtn:BitmapButton;
private var _selectedAllianceBtn:BitmapButton;

private var _infoDisplay:AllianceInfoDisplay;
private var _memberDisplay:AllianceMemberDisplay;

private var _allianceKey:String;

private var _infoText:String = 'CodeString.Alliance.Info'; //INFO
private var _memberText:String = 'CodeString.Alliance.Members'; //MEMBERS
private var _editMotdText:String = 'CodeString.Alliance.EditMotd'; //EDIT MOTD
private var _editDescriptionText:String = 'CodeString.Alliance.EditDescription'; //EDIT
DESCRIPTION

private var _promoteTitleText:String = 'CodeString.Alliance.Promote.Title'; //PROMOTE
PLAYER
private var _promoteBodyText:String = 'CodeString.Alliance.Promote.Body'; //Are you sure you
want to transfer leadership to this player?
private var _promoteText:String = 'CodeString.Alliance.Promote.Promote'; //PROMOTE
private var _noText:String = 'CodeString.Alliance.Promote.No'; //NO
private var _noAllianceFound:String = 'CodeString.Alliance.NoAllianceFound'; //Alliance not
found

[PostConstruct]
override public function init():void
{
super.init();

var windowBGClass:Class = Class(getDefinitionByName('AllianceMainBGBMD'));
_bg = new Bitmap(BitmapData(new windowBGClass()));

_infoBtn = ButtonFactory.getBitmapButton('AllianceSideBtnUpBMD', 17, 65, _infoText,
0xc9e6f6, 'AllianceSideBtnRollOverBMD', 'AllianceSideBtnDownBMD', null,
'AllianceSideBtnSelectedBMD');
```

```
_infoBtn.fontSize = 20;
_infoBtn.label.x -= 29;
_infoBtn.label.y += 3;
_infoBtn.selectable = true;
_infoBtn.selected = true;
_infoBtn.enabled = false;
selectBitmapButton(_infoBtn);
addListener(_infoBtn, MouseEvent.MOUSE_UP, onButtonClick);

_membersBtn = ButtonFactory.getBitmapButton('AllianceSideBtnUpBMD', 17, 135,
_memberText, 0xc9e6f6, 'AllianceSideBtnRollOverBMD', 'AllianceSideBtnDownBMD', null,
'AllianceSideBtnSelectedBMD');
_membersBtn.fontSize = 20;
_membersBtn.label.x -= 29;
_membersBtn.label.y += 3;
_membersBtn.selectable = true;
_membersBtn.enabled = false;
addListener(_membersBtn, MouseEvent.MOUSE_UP, onButtonClick);

_infoDisplay = new AllianceInfoDisplay();
_infoDisplay.onMotdUpdated = onUpdateMOTD;
_infoDisplay.onDescriptionUpdated = onUpdateDescription;
_infoDisplay.onPublicChanged = onPublicChanged;
_infoDisplay.onLeaveAlliance = onLeaveAlliance;
_infoDisplay.onSendAllianceMail = onSendAllianceMail;
_infoDisplay.x = 194;
_infoDisplay.y = 47;
_infoDisplay.setEnabled(false);
_infoDisplay.setVisible(false);
presenter.injectObject(_infoDisplay);

_memberDisplay = new AllianceMemberDisplay();
_memberDisplay.onPromoteMember = onPromoteMember;
_memberDisplay.onDemoteMember = onDemoteMember;
_memberDisplay.onRemoveMember = onRemoveMember;
_memberDisplay.onShowProfile = onShowProfile;
_memberDisplay.x = 194;
_memberDisplay.y = 47;
_memberDisplay.visible = false;

_title = new Label(22, 0xf0f0f0, 300, 30, true);
_title.constrictTextToSize = false;
_title.allCaps = true;
_title.align = TextFormatAlign.LEFT;
_title.x = 29;
_title.y = 10;
_title.text = _noAllianceFound;

presenter.addOnAllianceMembersUpdatedListener(onMembersUpdated);
presenter.addOnAllianceUpdatedListener(onAllianceUpdate);
```

```
presenter.addOnGenericAllianceMessageRecievedListener(handleAllianceMessage);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

addChild(_bg);
addChild(_closeBtn);
addChild(_title);
addChild(_infoBtn);
addChild(_membersBtn);
addChild(_infoDisplay);
addChild(_memberDisplay);

_infoBtn.visible = false;
_membersBtn.visible = false;
_infoDisplay.visible = false;

addEffects();
effectsIN();

if (_allianceKey != '')
{
_memberDisplay.allianceKey = _allianceKey;
presenter.allianceBaselineRequest(_allianceKey);
presenter.allianceRosterRequest(_allianceKey);
}
}

private function onButtonClick( e:MouseEvent ):void
{
if (_selectedAllianceBtn)
{
switch (_selectedAllianceBtn)
{
case _infoBtn:
if (e.target != _infoBtn)
{
_infoDisplay.visible = false;
unselectBitmapButton(_infoBtn);
}
break;
case _membersBtn:
if (e.target != _membersBtn)
{
_memberDisplay.visible = false;
unselectBitmapButton(_membersBtn);
}
break;
}

}
```

```
switch (e.target)
{
case _infoBtn:
if (_selectedAllianceBtn != _infoBtn)
{
_infoDisplay.visible = true;
selectBitmapButton(_infoBtn);
}
break;
case _membersBtn:
if (_selectedAllianceBtn != _membersBtn)
{
_memberDisplay.visible = true;
presenter.allianceRosterRequest(_allianceKey);
selectBitmapButton(_membersBtn);
}
break;
}
}

private function unselectBitmapButton( btn:BitmapButton ):void
{
btn.selectable = true;
btn.selected = false;
}

private function selectBitmapButton( btn:BitmapButton ):void
{
btn.selected = true;
btn.selectable = false;
_selectedAllianceBtn = btn;
}

private function onUpdateDescription( description:String ):void
{
var allianceEditInfoView:AllianceEditInfoView =
AllianceEditInfoView(_viewFactory.createView(AllianceEditInfoView));
allianceEditInfoView.titleText = _editDescriptionText;
allianceEditInfoView.maxChars = 512;
allianceEditInfoView.bodyText = description;
allianceEditInfoView.callback = presenter.allianceSetDescription;
_viewFactory.notify(allianceEditInfoView);
}

private function onUpdateMOTD( motd:String ):void
{
var allianceEditInfoView:AllianceEditInfoView =
AllianceEditInfoView(_viewFactory.createView(AllianceEditInfoView));
allianceEditInfoView.titleText
```

```
= _editMotdText;
allianceEditInfoView.maxChars = 256;
allianceEditInfoView.bodyText = motd;
allianceEditInfoView.callback = presenter.allianceSetMOTD;
_viewFactory.notify(allianceEditInfoView);
}

private function onPublicChanged( isPublic:Boolean ):void
{
presenter.allianceSetPublic(isPublic);
}

private function onPromoteMember( v:AllianceMemberVO ):void
{

if (v.rank == AllianceRankEnum.OFFICER)
{
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_promoteText, presenter.alliancePlayerPromote, [v.key],
true, ButtonEnum.GREEN_A));
buttons.push(new ButtonPrototype(_noText));
showConfirmation(_promoteTitleText, _promoteBodyText, buttons);
} else
presenter.alliancePlayerPromote(v.key);
}

private function onDemoteMember( v:AllianceMemberVO ):void
{
presenter.alliancePlayerDemote(v.key);
}

private function onRemoveMember( v:AllianceMemberVO ):void
{
presenter.alliancePlayerKick(v.key);
}

private function onShowProfile( v:AllianceMemberVO ):void
{
var playerProfileView:PlayerProfileView =
PlayerProfileView(_viewFactory.createView(PlayerProfileView));
playerProfileView.playerKey = v.key;
_viewFactory.notify(playerProfileView);
}

private function onSendAllianceMail():void
{
var newMailView:NewMailView = NewMailView(_viewFactory.createView(NewMailView));
newMailView.setMessageInfo('Alliance', '');
_viewFactory.notify(newMailView);
}

private
```

```
function onLeaveAlliance():void
{
presenter.allianceLeave();
destroy();
}

public function handleAllianceMessage( messageEnum:int, allianceKey:String ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.SET_SUCCESS:
if (_allianceKey != '')
{
presenter.allianceRosterRequest(_allianceKey);
presenter.allianceBaselineRequest(_allianceKey);
}
break;
case AllianceResponseEnum.KICKED:
destroy();
break;
}
}

private function onMembersUpdated( allianceKey:String, v:Vector.<AllianceMemberVO> ):void
{
if (allianceKey == _allianceKey && _memberDisplay)
_memberDisplay.onMembersUpdated(v);
}

private function onAllianceUpdate( allianceKey:String, v:AllianceVO ):void
{
if (allianceKey == _allianceKey)
{
if (_infoBtn)
_infoBtn.enabled = true;
if (_membersBtn)
_membersBtn.enabled = true;
if (_title)
_title.text = v.name;

if (_infoDisplay)
{
_infoDisplay.setEnabled(true);
_infoDisplay.setVisible(true);
_infoDisplay.onAllianceUpdated(v);
}
_infoBtn.visible = true;
_membersBtn.visible = true;

if (_selectedAllianceBtn == _infoBtn)
{
```

```
_infoDisplay.visible = true;
}
else if (_selectedAllianceBtn == _membersBtn)
{
_memberDisplay.visible = true;
}
}
}

public function set allianceKey( v:String ):void { _allianceKey = v; }

override public function get width():Number { return _bg.width; }
override public function get height():Number { return _bg.height; }

[Inject]
public function set presenter( v:IAlliancePresenter ):void { _presenter = v; }
public function get presenter():IAlliancePresenter { return IAlliancePresenter(_presenter); }

override public function destroy():void
{
presenter.removeOnAllianceMembersUpdatedListener(onMembersUpdated);
presenter.removeOnAllianceUpdatedListener(onAllianceUpdate);
presenter.removeOnGenericAllianceMessageRecievedListener(handleAllianceMessage);
super.destroy();

_bg = null;
_currentState = null;

if (_title)
_title.destroy();

_title = null;

if (_infoBtn)
_infoBtn.destroy();

_infoBtn = null;

if (_membersBtn)
_membersBtn.destroy();

_membersBtn = null;

if (_selectedAllianceBtn)
_selectedAllianceBtn.destroy();

_selectedAllianceBtn = null;

if (_infoDisplay)
_infoDisplay.destroy();
```

```
_infoDisplay = null;

if (_memberDisplay)
_memberDisplay.destroy();

_memberDisplay = null;

_allianceKey = '';
}
}
}
```

File 782: igw\com\ui\modal\alliances\noalliance\AllianceEntry.as

```
package com.ui.modal.alliances.noalliance
{
import com.model.alliance.AllianceVO;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;

public class AllianceEntry extends Sprite
{
public var onViewClick:Signal;
public var onAcceptClick:Signal;

public static var TYPE_OPEN_ALLIANCE:uint = 0;
public static var TYPE_ALLIANCE_INVITE:uint = 1;

private var _bg:Bitmap;

private var _name:Label;
private var _members:Label;

private var _joinBtn:BitmapButton;
private var _viewBtn:BitmapButton;

private var _alliance:AllianceVO;

private
```

```
var _isInvite:Boolean;

private var _acceptBtnText:String = 'CodeString.Shared.Accept'; //ACCEPT
private var _joinBtnText:String = 'CodeString.Shared.Join'; //JOIN
private var _viewBtnText:String = 'CodeString.Alliance.View' //VIEW
private var _outOf:String = 'CodeString.Shared.OutOf'
//[[Number.MinValue]]/[[Number.MaxValue]]

public function AllianceEntry( alliance:AllianceVO, index:uint, type:uint )
{
onViewClick = new Signal(String);
onAcceptClick = new Signal(String);

super();

if ((index + 1) % 2 == 0)
{
_bg = PanelFactory.getScaleBitmapPanel('AllianceMemberRowBMD', 497, 30, new
Rectangle(15, 11, 2, 2));
}

_isInvite = (type == TYPE_ALLIANCE_INVITE) ? true : false;

_name = new Label(16, 0xf0f0f0, 361, 39, false);
_name.constrictTextToSize = false;
_name.align = TextFormatAlign.LEFT;

_members = new Label(16, 0xf0f0f0, 48, 39, false);
_members.constrictTextToSize = false;
_members.constrictTextToSize = false;
_members.align = TextFormatAlign.CENTER;

_viewBtn = ButtonFactory.getBitmapButton('BlueBtnCNeutralBMD', 0, 0, _viewBtnText, 0xf0f0f0,
'BlueBtnCRollOverBMD', 'BlueBtnCSelectedBMD', null, 'BlueBtnCSelectedBMD');
_viewBtn.scaleX = 0.5;
_viewBtn.scaleY = 0.75;
_viewBtn.addEventListener(MouseEvent.CLICK, onViewAllianceClick, false, 0, true);

_joinBtn = ButtonFactory.getBitmapButton('BlueBtnCNeutralBMD', 0, 0, (_isInvite) ?
_acceptBtnText : _joinBtnText, 0xf0f0f0, 'BlueBtnCRollOverBMD', 'BlueBtnCSelectedBMD', null,
'BlueBtnCSelectedBMD');
_joinBtn.scaleX = 0.5;
_joinBtn.scaleY = 0.75;
_joinBtn.addEventListener(MouseEvent.CLICK, onAcceptAllianceClick, false, 0, true);

if ((index + 1) % 2 == 0)
addChild(_bg);

addChild(_name);
addChild(_members);
addChild(_viewBtn);
```

```
addChild(_joinBtn);

update(alliance);
}

private function onViewAllianceClick( e:MouseEvent ):void
{
if (_alliance)
onViewClick.dispatch(_alliance.key);
}

private function onAcceptAllianceClick( e:MouseEvent ):void
{
if (_alliance)
onAcceptClick.dispatch(_alliance.key);
}

public function update( alliance:AllianceVO ):void
{
_alliance = alliance;

_name.text = _alliance.name;
_members.setTextWithTokens(_outOf, {'[[Number.MinValue]]':alliance.memberCount,
'[[Number.MaxValue]]':100});
layout();
}

private function layout():void
{
var center:Number = 30 * 0.5 - 11;

_name.x = 5;
_name.y = center;

_members.x = 293;
_members.y = center;

_viewBtn.x = 343;
_viewBtn.y = 5;

_joinBtn.x = _viewBtn.x + _viewBtn.width + 5;
_joinBtn.y = 5;
}

public function set enabled( v:Boolean ):void
{
_joinBtn.enabled = v;
}

public
```

```
function get filterBy():String { return _alliance.name.toLowerCase(); }

public function get memberCount():int { return _alliance.memberCount; }

override public function get height():Number { return 30; }
override public function get width():Number { return 497; }

public function destroy():void
{
if (onAcceptClick)
onAcceptClick.removeAll();

onAcceptClick = null;

if (onViewClick)
onViewClick.removeAll();

onViewClick = null;

_bg = null;

if (_name)
_name.destroy();
_name = null;

if (_name)
_name.destroy();
_name = null;

if (_members)
_members.destroy();
_members = null;


if (_viewBtn)
_viewBtn.destroy();
_viewBtn = null;

if (_joinBtn)
_joinBtn.destroy();
_joinBtn = null;

}
}
}

------------------------------------
File 783: igw\com\ui\modal\alliances\noalliance\AllianceInvitesDisplay.as
package com.ui.modal.alliances.noalliance
{
import
```

```
com.enum.server.AllianceResponseEnum;
import com.model.alliance.AllianceInviteVO;
import com.model.alliance.AllianceVO;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.label.Label;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

public class AllianceInvitesDisplay extends Sprite
{
private var _title:Label;

private var _alliancesBG:Bitmap;

private var _allianceHolder:Sprite;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _alliances:Vector.<AllianceEntry>;

public var joinAllianceClick:Function;
public var viewAllianceClick:Function;

private var _pendingInvites:String = 'CodeString.Alliance.PendingInvites'; //Pending Invites

public function AllianceInvitesDisplay()
{
super();

_alliances = new Vector.<AllianceEntry>;

_title = new Label(20, 0xf0f0f0, 150, 25);
_title.align = TextFormatAlign.LEFT;
_title.allCaps = true;
_title.x = 3;
_title.y = 45;
_title.text = _pendingInvites;

_alliancesBG = _alliancesBG = PanelFactory.getPanel('AllianceOpenBGBMD');
_alliancesBG.x = 3;
_alliancesBG.y = 70;

_allianceHolder
```

```
= new Sprite();
_allianceHolder.x = _alliancesBG.x;
_allianceHolder.y = _alliancesBG.y;
_maxHeight = 0;

_scrollRect = new Rectangle(_allianceHolder.x, _allianceHolder.y, 497, 290);
_scrollRect.y = 0;
_allianceHolder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _alliancesBG.x + _alliancesBG.width;
var scrollbarYPos:Number = _alliancesBG.y;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this, _allianceHolder);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 29;

addChild(_title);
addChild(_alliancesBG);
addChild(_allianceHolder);
addChild(_scrollbar);
}

private function onJoinAllianceClick( allianceKey:String ):void
{

if (joinAllianceClick != null)
{
enableJoining(false);
joinAllianceClick(allianceKey);
}
}

private function onViewAllianceClick( allianceKey:String ):void
{
if (viewAllianceClick != null)
viewAllianceClick(allianceKey);
}

public function handleAllianceMessage( messageEnum:int ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.JOIN_FAILED_TOOMANYPLAYERS:
enableJoining(true);
break;
}
}
```

```
public function enableJoining( v:Boolean ):void
{
var len:uint = _alliances.length;
var currentEntry:AllianceEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _alliances[i];
if (currentEntry)
currentEntry.enabled = v;

}
}

private function memberSortingFunction(entryA:AllianceEntry, entryB:AllianceEntry):Number
{
if (entryA.memberCount.valueOf() > entryB.memberCount.valueOf())
{
return -1;
}
else if (entryA.memberCount.valueOf() < entryB.memberCount.valueOf())
{
return 1;
}
else
{
return 0;
}
}

public function onInvitesUpdated( v:Dictionary ):void
{
var i:uint = 0;
var currentEntryCount:uint = _alliances.length;
var currentEntry:AllianceEntry;
for each (var invite:AllianceInviteVO in v)
{
if (i < currentEntryCount)
{
currentEntry = _alliances[i];
currentEntry.update(invite.alliance);
} else
{
currentEntry = new AllianceEntry(invite.alliance, i + 1, AllianceEntry.TYPE_OPEN_ALLIANCE);
currentEntry.onAcceptClick.add(onJoinAllianceClick);
currentEntry.onViewClick.add(onViewAllianceClick);
_alliances.push(currentEntry);
}
_allianceHolder.addChild(currentEntry);
++i;
```

```
}

//sort alliances by member count
_alliances.sort(memberSortingFunction);

layout();
}

private function layout():void
{
var len:uint = _alliances.length;
var selection:AllianceEntry;
var yPos:int = 0;
var xPos:int = _allianceHolder.x;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _alliances[i];
selection.x = xPos;
selection.y = yPos;
_maxHeight += selection.height - 1;
yPos += selection.height - 1;
}
_scrollbar.updateScrollableHeight(_maxHeight);
}


private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_allianceHolder.scrollRect = _scrollRect;
}

public function destroy():void
{
_alliancesBG = null;
_allianceHolder = null;
_scrollRect = null;
joinAllianceClick = null;
viewAllianceClick = null;

if (_title)
_title.destroy();

_title = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar
```

```
= null;

var len:uint = _alliances.length;
var currentEntry:AllianceEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _alliances[i];
currentEntry.destroy();
currentEntry = null;
}
_alliances.length = 0;
}
}
}
```

File 784: igw\com\ui\modal\alliances\noalliance\CreateAllianceDisplay.as

```
package com.ui.modal.alliances.noalliance
{
import com.enum.server.AllianceResponseEnum;
import com.service.language.Localization;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.events.TextEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

public class CreateAllianceDisplay extends Sprite
{
public var createAllianceClick:Function;

private var _title:Label;

private var _name:Label;
private var _description:Label;
private var _checkboxText:Label;

private var _nameBG:Bitmap;
private var _descriptionBG:ScaleBitmap;

private
```

```
var _descriptionHolder:Sprite;

private var _publicCheckbox:BitmapButton;
private var _createAllianceBtn:BitmapButton;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _tooltips:Tooltips;

private const ALLIANCE_MAX_NAME_LENGTH:uint = 25;

private var _allianceCreateText:String = 'CodeString.Alliance.Create'; //CREATE
private var _allianceDescriptionText:String = 'CodeString.Alliance.Description'; //Alliance
Description
private var _allianceNameText:String = 'CodeString.Alliance.AllianceName'; // Alliance Name
private var _allianceCreatingStateText:String = 'CodeString.Alliance.Creating'; //CREATING
private var _allianceCreateTitleText:String = 'CodeString.Alliance.CreateAlliance'; //Create
Alliance
private var _alliancePublicText:String = 'CodeString.Alliance.Public'; //Public
private var _alliancePublicDescriptionText:String = 'CodeString.Alliance.PublicDescription';
//Public alliances can be joined by anyone in your faction!


public function CreateAllianceDisplay()
{
super();

_nameBG = PanelFactory.getScaleBitmapPanel('AllianceMemberRowBMD', 361, 24, new
Rectangle(15, 11, 2, 2));
_nameBG.x = 27;
_nameBG.y = 90;

_descriptionBG = PanelFactory.getScaleBitmapPanel('AllianceMemberRowBMD', 460, 114, new
Rectangle(15, 11, 2, 2));
_descriptionBG.x = 27;
_descriptionBG.y = _nameBG.y + _nameBG.height + 7;

_title = new Label(20, 0xf0f0f0, 150, 25);
_title.align = TextFormatAlign.LEFT;
_title.allCaps = true;
_title.x = 27;
_title.y = 65;
_title.text = _allianceCreatingStateText;

_name = new Label(18, 0xa9dcff, 469, 30);
_name.align = TextFormatAlign.LEFT;
_name.text = _allianceNameText;
_name.x
```

```
= _nameBG.x + 8;
_name.y = _nameBG.y;
_name.maxChars = ALLIANCE_MAX_NAME_LENGTH;
_name.multiline = true;
_name.allowInput = true;
_name.clearOnFocusIn = true;
_name.letterSpacing = .8;
_name.restrict = "A-Za-z0-9'_ ";
_name.addLabelColor(0xbdfefd, 0x000000);

_description = new Label(18, 0xa9dcff, 450, 235);
_description.align = TextFormatAlign.LEFT;
_description.text = _allianceDescriptionText;
_description.maxChars = 512;
_description.multiline = true;
_description.allowInput = true;
_description.clearOnFocusIn = true;
_description.letterSpacing = .8;
_description.addLabelColor(0xbdfefd, 0x000000);
_description.addEventListener(Event.CHANGE, onTextUpdated, false, 0, true);

_descriptionHolder = new Sprite();
_descriptionHolder.x = _descriptionBG.x + 5;
_descriptionHolder.y = _descriptionBG.y + 3;
_descriptionHolder.addChild(_description);

_scrollRect = new Rectangle(0, 0, _descriptionHolder.width, _descriptionBG.height - 5);
_scrollRect.y = 0;
_descriptionHolder.scrollRect = _scrollRect;

_publicCheckbox = ButtonFactory.getBitmapButton('CheckboxBtnUncheckedBMD', 0, 0, '', 0,
null, 'CheckboxBtnUncheckedBMD', null, 'CheckboxBtnCheckedBMD');
_publicCheckbox.x = _nameBG.x + _nameBG.width + 5;
_publicCheckbox.y = _nameBG.y + 1;
_publicCheckbox.selectable = true;
_publicCheckbox.selected = true;

_checkboxText = new Label(18, 0xa9dcff, 469, 30);
_checkboxText.align = TextFormatAlign.LEFT;
_checkboxText.text = _alliancePublicText;
_checkboxText.x = _publicCheckbox.x + _publicCheckbox.width + 5;
_checkboxText.y = _publicCheckbox.y + (_checkboxText.textHeight - _publicCheckbox.height) *
0.5;

_createAllianceBtn = ButtonFactory.getBitmapButton('BlueBtnCNeutralBMD', 339, 565,
_allianceCreateText, 0xf0f0f0, 'BlueBtnCRollOverBMD', 'BlueBtnCSelectedBMD', null,
'BlueBtnCSelectedBMD');
_createAllianceBtn.x = _descriptionBG.x + (_descriptionBG.width - _createAllianceBtn.width) *
0.5;
_createAllianceBtn.y = _descriptionBG.y + _descriptionBG.height + 5;
_createAllianceBtn.addEventListener(MouseEvent.CLICK,
```

```
onCreateAllianceClick, false, 0, true);

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _descriptionBG.x + _descriptionBG.width + 5;
var scrollbarYPos:Number = _descriptionBG.y;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this, _descriptionHolder);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 7;

addChild(_title);
addChild(_nameBG);
addChild(_descriptionBG);
addChild(_name);
addChild(_descriptionHolder);
addChild(_scrollbar);
addChild(_publicCheckbox);
addChild(_checkboxText);
addChild(_createAllianceBtn);
}

private function onCreateAllianceClick( e:MouseEvent ):void
{
var name:String = _name.text;
name.split(' ').join('');
if (name.length > 0 && name != '' && name != _name.inputMessage)
{
if (createAllianceClick != null)
{
var description:String = (_description.text == _description.inputMessage) ? '' : _description.text;
createAllianceClick(name, _publicCheckbox.selected, _description.text);
_createAllianceBtn.enabled = false;
_createAllianceBtn.text = _allianceCreatingStateText;
}

}
}

public function handleAllianceMessage( messageEnum:int ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_NAMEINUSE:
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_UNKNOWN:
_createAllianceBtn.enabled = true;
_createAllianceBtn.text = _allianceCreateText;
break;
}
```

```
}

private function onTextUpdated( e:Event ):void
{
_scrollbar.updateScrollableHeight(_description.textHeight);

var car:int = _description.caretIndex;
var rect:Rectangle = _description.getCharBoundaries(car - 1);

if (rect != null)
{
if (rect.y + rect.height > _scrollRect.height)
{
var percent:Number = ((rect.y + rect.height) / _description.textHeight)
_scrollbar.updateScrollPercent(percent);
} else if (rect.y + rect.height < _scrollRect.height && _scrollbar.percent != 0)
_scrollbar.updateScrollPercent(0);
}
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_description.textHeight - _scrollRect.height) * percent;
_descriptionHolder.scrollRect = _scrollRect;
}

[Inject]
public function set tooltips( value:Tooltips ):void
{
_tooltips = value;

if (_publicCheckbox)
_tooltips.addTooltip(_publicCheckbox, this, null,
Localization.instance.getString(_alliancePublicDescriptionText));
}

public function destroy():void
{
if (_tooltips)
_tooltips.removeTooltip(null, this);

_tooltips = null;

_nameBG = null;
_descriptionBG = null;
_descriptionHolder = null;
_scrollRect = null;
createAllianceClick = null;

if
```

```
(_title)
_title.destroy();

_title = null;

if (_name)
_name.destroy();

_name = null;

if (_description)
_description.destroy();

_description = null;

if (_checkboxText)
_checkboxText.destroy();

_checkboxText = null;

if (_publicCheckbox)
_publicCheckbox.destroy();

_publicCheckbox = null;

if (_createAllianceBtn)
_createAllianceBtn.destroy();

_createAllianceBtn = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar = null;
}
}
}

------------------------------------
File 785: igw\com\ui\modal\alliances\noalliance\JoinAllianceDisplay.as
package com.ui.modal.alliances.noalliance
{
import com.Application;
import com.controller.keyboard.KeyboardController;
import com.controller.keyboard.KeyboardKey;
import com.enum.server.AllianceResponseEnum;
import com.model.alliance.AllianceVO;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import
```

```
com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.display.Stage;
import flash.events.Event;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

public class JoinAllianceDisplay extends Sprite
{
public var joinAllianceClick:Function;
public var viewAllianceClick:Function;

private var _title:Label;
private var _search:Label;

private var _alliancesBG:Bitmap;
private var _searchBG:Bitmap;

private var _searchBtn:BitmapButton;

private var _allianceHolder:Sprite;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _alliances:Vector.<AllianceEntry>;
private var _visibleAlliances:Vector.<AllianceEntry>;

private var _stage:Stage;
private var _keyboard:KeyboardController;

private var _searchAlliances:String = 'CodeString.Shared.SearchAlliances'; //Search Alliances....
private var _publicAlliances:String = 'CodeString.JoinAlliance.PublicAlliances'; //PUBLIC
ALLIANCES

public function JoinAllianceDisplay()
{
super();

_stage = Application.STAGE;

_alliances = new Vector.<AllianceEntry>;
_visibleAlliances = new Vector.<AllianceEntry>;

_searchBG
```

```
= PanelFactory.getPanel('AllianceMemberSearchBMD');
_searchBG.x = 273;
_searchBG.y = 24;

_title = new Label(20, 0xf0f0f0, 150, 25);
_title.align = TextFormatAlign.LEFT;
_title.x = 3;
_title.y = 45;
_title.text = _publicAlliances;

_alliancesBG = PanelFactory.getPanel('AllianceOpenBGBMD');
_alliancesBG.x = 3;
_alliancesBG.y = 70;

_search = new Label(18, 0xf0f0f0, 184, 30, true);
_search.align = TextFormatAlign.LEFT;
_search.text = _searchAlliances;
_search.x = _searchBG.x + 8;
_search.y = _searchBG.y;
_search.maxChars = 20;
_search.allowInput = true;
_search.clearOnFocusIn = true;
_search.letterSpacing = .8;
_search.addLabelColor(0xbdfefd, 0x000000);
_search.addEventListener(Event.CHANGE, onChanged, false, 0, true);

_searchBtn = ButtonFactory.getBitmapButton('AllianceSearchBtnUpBMD', _searchBG.x + 194,
_searchBG.y + 3, '', 0, 'AllianceSearchBtnRollOverBMD');

_allianceHolder = new Sprite();
_allianceHolder.x = _alliancesBG.x;
_allianceHolder.y = _alliancesBG.y;
_maxHeight = 0;

_scrollRect = new Rectangle(_allianceHolder.x, _allianceHolder.y, 497, 290);
_scrollRect.y = 0;
_allianceHolder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _alliancesBG.x + _alliancesBG.width;
var scrollbarYPos:Number = _alliancesBG.y;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this, _allianceHolder);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 29;

addChild(_title);
addChild(_alliancesBG);
```

```
addChild(_allianceHolder);
addChild(_scrollbar);
addChild(_searchBG);
addChild(_search);
addChild(_searchBtn);
}

public function handleAllianceMessage( messageEnum:int ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.JOIN_FAILED_TOOMANYPLAYERS:
enableJoining(true);
break;
}
}

public function enableJoining( v:Boolean ):void
{
var len:uint = _alliances.length;
var currentEntry:AllianceEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _alliances[i];
if (currentEntry)
currentEntry.enabled = v;

}
}

public function onAlliancesUpdated( v:Dictionary ):void
{
var i:uint = 0;
var currentEntryCount:uint = _alliances.length;
var currentEntry:AllianceEntry;
for each (var alliance:AllianceVO in v)
{
if (i < currentEntryCount)
{
currentEntry = _alliances[i];
currentEntry.update(alliance);
} else
{
currentEntry = new AllianceEntry(alliance, i + 1, AllianceEntry.TYPE_OPEN_ALLIANCE);
currentEntry.onAcceptClick.add(onJoinAllianceClick);
currentEntry.onViewClick.add(onViewAllianceClick);
_alliances.push(currentEntry);
}
_allianceHolder.addChild(currentEntry);
++i;
```

```
}
onChanged();
layout();
}

private function onJoinAllianceClick( allianceKey:String ):void
{

if (joinAllianceClick != null)
{
enableJoining(false);
joinAllianceClick(allianceKey);
}
}

private function onViewAllianceClick( allianceKey:String ):void
{
if (viewAllianceClick != null)
viewAllianceClick(allianceKey);
}



private function layout():void
{
var len:uint = _visibleAlliances.length;
var selection:AllianceEntry;
var yPos:int = 0;
var xPos:int = _allianceHolder.x;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _visibleAlliances[i];
selection.x = xPos;
selection.y = yPos;
_maxHeight += selection.height - 1;
yPos += selection.height - 1;
}
_scrollbar.updateScrollableHeight(_maxHeight);
}


private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_allianceHolder.scrollRect = _scrollRect;
}

private function memberSortingFunction(entryA:AllianceEntry, entryB:AllianceEntry):Number
{
```

```
if (entryA.memberCount.valueOf() > entryB.memberCount.valueOf())
{
return -1;
}
else if (entryA.memberCount.valueOf() < entryB.memberCount.valueOf())
{
return 1;
}
else
{
return 0;
}
}

private function filterAlliances( filterBy:String ):void
{
if (_allianceHolder.numChildren > 0)
_allianceHolder.removeChildren(0, (_allianceHolder.numChildren - 1));

_visibleAlliances.length = 0;

var len:uint = _alliances.length;
var currentEntry:AllianceEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _alliances[i];
if (filterBy == '' || filterBy == _search.inputMessage.toLowerCase() ||
currentEntry.filterBy.indexOf(filterBy) != -1)
{
_allianceHolder.addChild(currentEntry);
_visibleAlliances.push(currentEntry);
}
}

//sort alliances by member count
_visibleAlliances.sort(memberSortingFunction);

layout();
_scrollbar.resetScroll();
}

private function onChanged( e:Event = null ):void
{
filterAlliances(_search.text.toLowerCase());
}

private function onEnterPress( keyCode:uint ):void
{
if (_stage.focus == _search)
{
```

```
filterAlliances(_search.text.toLowerCase());
addEventListener(Event.ENTER_FRAME, removeFocus, false, 0, true);
}
}

private function removeFocus( e:Event ):void
{
removeEventListener(Event.ENTER_FRAME, removeFocus);
if (_stage)
_stage.focus = Application.STAGE;
}

[Inject]
public function set keyboard( value:KeyboardController ):void
{
_keyboard = value;
_keyboard.addKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);
}


public function destroy():void
{
_alliancesBG = null;
_searchBG = null;
_allianceHolder = null;
_scrollRect = null;
_stage = null;
joinAllianceClick = null;
viewAllianceClick = null;

if (_title)
_title.destroy();

_title = null;

if (_search)
_search.destroy();

_search = null;

if (_searchBtn)
_searchBtn.destroy();

_searchBtn = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar = null;

_visibleAlliances.length
```

```
= 0;

var len:uint = _alliances.length;
var currentEntry:AllianceEntry;
for (var i:uint = 0; i < len; ++i)
{
currentEntry = _alliances[i];
currentEntry.destroy();
currentEntry = null;
}
_alliances.length = 0;

if (_keyboard)
_keyboard.removeKeyUpListener(onEnterPress, KeyboardKey.ENTER.keyCode);

_keyboard = null;
}
}
}

-----------------------------------
File 786: igw\com\ui\modal\alliances\noalliance\NoAllianceView.as
package com.ui.modal.alliances.noalliance
{
import com.enum.server.AllianceResponseEnum;
import com.model.player.CurrentUser;
import com.presenter.shared.IAlliancePresenter;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.alliances.alliance.AllianceView;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

public class NoAllianceView extends View
{
private var _bg:Bitmap;
private var _closeBtn:BitmapButton;
private var _currentState:Sprite;

private var _title:Label;

private var _joinDisplay:JoinAllianceDisplay;
private
```

```
var _invitesDisplay:AllianceInvitesDisplay;
private var _createDisplay:CreateAllianceDisplay;

private var _joinBtn:BitmapButton;
private var _inviteBtn:BitmapButton;
private var _createBtn:BitmapButton;
private var _selectedAllianceBtn:BitmapButton;

private var _allianceText:String = 'CodeString.Shared.Alliances'; //ALLIANCES
private var _allianceJoinText:String = 'CodeString.NoAlliance.Join'; //JOIN
private var _allianceInvitesText:String = 'CodeString.NoAlliance.Invites'; //INVITES
private var _allianceCreateText:String = 'CodeString.Alliance.Create'; //CREATE

[PostConstruct]
override public function init():void
{
super.init();

var windowBGClass:Class = Class(getDefinitionByName('AllianceMainBGBMD'));
_bg = new Bitmap(BitmapData(new windowBGClass()));

_joinBtn = ButtonFactory.getBitmapButton('AllianceSideBtnUpBMD', 17, 64, _allianceJoinText,
0xc9e6f6, 'AllianceSideBtnRollOverBMD', 'AllianceSideBtnDownBMD', null,
'AllianceSideBtnSelectedBMD');
_joinBtn.fontSize = 20;
_joinBtn.label.x -= 29;
_joinBtn.label.y += 3;
_joinBtn.selectable = true;
addListener(_joinBtn, MouseEvent.MOUSE_UP, onButtonClick);

_inviteBtn = ButtonFactory.getBitmapButton('AllianceSideBtnUpBMD', 17, 135,
_allianceInvitesText, 0xc9e6f6, 'AllianceSideBtnRollOverBMD', 'AllianceSideBtnDownBMD', null,
'AllianceSideBtnSelectedBMD');
_inviteBtn.fontSize = 20;
_inviteBtn.label.x -= 29;
_inviteBtn.label.y += 3;
_inviteBtn.selectable = true;
addListener(_inviteBtn, MouseEvent.MOUSE_UP, onButtonClick);

_createBtn = ButtonFactory.getBitmapButton('AllianceSideBtnUpBMD', 17, 205,
_allianceCreateText, 0xc9e6f6, 'AllianceSideBtnRollOverBMD', 'AllianceSideBtnDownBMD',
null, 'AllianceSideBtnSelectedBMD');
_createBtn.fontSize = 20;
_createBtn.label.x -= 29;
_createBtn.label.y += 3;
_createBtn.selectable = true;
addListener(_createBtn, MouseEvent.MOUSE_UP, onButtonClick);

_joinDisplay = new JoinAllianceDisplay();
_joinDisplay.joinAllianceClick = joinAlliance;
_joinDisplay.viewAllianceClick
```

```
= viewAlliance;
presenter.injectObject(_joinDisplay);
_joinDisplay.x = 194;
_joinDisplay.y = 47;
_joinDisplay.visible = false;

_invitesDisplay = new AllianceInvitesDisplay();
_invitesDisplay.joinAllianceClick = joinAlliance;
_invitesDisplay.viewAllianceClick = viewAlliance;
_invitesDisplay.x = 194;
_invitesDisplay.y = 47;
_invitesDisplay.visible = false;

_createDisplay = new CreateAllianceDisplay();
_createDisplay.createAllianceClick = createAlliance;
presenter.injectObject(_createDisplay);
_createDisplay.x = 194;
_createDisplay.y = 47;
_createDisplay.visible = false;

_title = new Label(20, 0xf0f0f0, 150, 25);
_title.align = TextFormatAlign.LEFT;
_title.x = 29;
_title.y = 10;
_title.text = _allianceText;

presenter.addOnOpenAlliancesUpdatedListener(onAlliancesUpdated);
presenter.addOnInvitedAlliancesUpdatedListener(onInvitedAlliancesUpdated);
presenter.addOnGenericAllianceMessageRecievedListener(handleAllianceMessage);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

addChild(_bg);
addChild(_closeBtn);
addChild(_title);
addChild(_createBtn);
addChild(_joinBtn);
addChild(_inviteBtn);
addChild(_joinDisplay);
addChild(_invitesDisplay);
addChild(_createDisplay);

addEffects();
effectsIN();

presenter.alliancePublicAllianceRequest();

_joinDisplay.visible = true;
selectBitmapButton(_joinBtn);
}
```

```
private function onButtonClick( e:MouseEvent ):void
{
if (_selectedAllianceBtn)
{
switch (_selectedAllianceBtn)
{
case _joinBtn:
if (e.target != _joinBtn)
{
_joinDisplay.visible = false;
unselectBitmapButton(_joinBtn);
}
break;
case _inviteBtn:
if (e.target != _inviteBtn)
{
_invitesDisplay.visible = false;
unselectBitmapButton(_inviteBtn);
}
break;
case _createBtn:
if (e.target != _createBtn)
{
_createDisplay.visible = false;
unselectBitmapButton(_createBtn);
}
break;
}
}

switch (e.target)
{
case _joinBtn:
if (_selectedAllianceBtn != _joinBtn)
{
presenter.alliancePublicAllianceRequest();
_joinDisplay.visible = true;
selectBitmapButton(_joinBtn);
}
break;
case _inviteBtn:
if (_selectedAllianceBtn != _inviteBtn)
{
if (presenter)
_invitesDisplay.onInvitesUpdated(presenter.getAllianceInvites());

_invitesDisplay.visible = true;
selectBitmapButton(_inviteBtn);
}
```

```
break;
case _createBtn:
if (_selectedAllianceBtn != _createBtn)
{
_createDisplay.visible = true;
selectBitmapButton(_createBtn);
}
break;
}
}

private function unselectBitmapButton( btn:BitmapButton ):void
{
btn.selectable = true;
btn.selected = false;
}

private function selectBitmapButton( btn:BitmapButton ):void
{
btn.selected = true;
btn.selectable = false;
_selectedAllianceBtn = btn;
}

private function createAlliance( name:String, isPublic:Boolean, description:String ):void
{
presenter.allianceCreateRequest(name, isPublic, description);
}

private function joinAlliance( allianceKey:String ):void
{
presenter.allianceJoin(allianceKey);
}

private function viewAlliance( allianceKey:String ):void
{
if (allianceKey != '')
{
var allianceView:AllianceView = AllianceView(_viewFactory.createView(AllianceView));
allianceView.allianceKey = allianceKey;
_viewFactory.notify(allianceView);
}
}

private function onAlliancesUpdated( v:Dictionary ):void
{
if (_joinDisplay)
_joinDisplay.onAlliancesUpdated(v);
}

private
```

```
function onInvitedAlliancesUpdated( v:Dictionary ):void
{
if (_invitesDisplay)
_invitesDisplay.onInvitesUpdated(v);
}

public function handleAllianceMessage( messageEnum:int, allianceKey:String ):void
{
switch (messageEnum)
{
case AllianceResponseEnum.ALLIANCE_CREATED:
case AllianceResponseEnum.JOINED:
CurrentUser.alliance = allianceKey;
destroy();
break;
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_NAMEINUSE:
case AllianceResponseEnum.ALLIANCE_CREATION_FAILED_UNKNOWN:
_createDisplay.handleAllianceMessage(messageEnum);
break;
case AllianceResponseEnum.JOIN_FAILED_TOOMANYPLAYERS:
break;
}
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IAlliancePresenter ):void { _presenter = v; }
public function get presenter():IAlliancePresenter { return IAlliancePresenter(_presenter); }

override public function destroy():void
{
presenter.removeOnOpenAlliancesUpdatedListener(onAlliancesUpdated);
presenter.removeOnInvitedAlliancesUpdatedListener(onInvitedAlliancesUpdated);
presenter.removeOnGenericAllianceMessageRecievedListener(handleAllianceMessage);
super.destroy();

_bg = null;
_closeBtn.destroy();
_closeBtn = null;
_currentState = null;

if (_title)
_title.destroy();

_title = null;

if (_createDisplay)
_createDisplay.destroy();

_createDisplay
```

```
= null;

if (_joinDisplay)
_joinDisplay.destroy();

_joinDisplay = null;

if (_invitesDisplay)
_invitesDisplay.destroy();

_invitesDisplay = null;

if (_createBtn)
_createBtn.destroy();

_createBtn = null;

if (_joinBtn)
_joinBtn.destroy();

_joinBtn = null;

if (_inviteBtn)
_inviteBtn.destroy();

_inviteBtn = null;

if (_selectedAllianceBtn)
_selectedAllianceBtn.destroy();

_selectedAllianceBtn = null;
}
}
}

-----------------------------------
File 787: igw\com\ui\modal\battle\BattleEndParticipantDisplay.as
package com.ui.modal.battle
{
import com.enum.CategoryEnum;
import com.enum.TypeEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.battle.BattleEntityVO;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import
```

```
com.ui.modal.dock.ShipIcon;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.text.TextFormatAlign;

import org.shared.ObjectPool;

public class BattleEndParticipantDisplay extends Sprite
{
private var _player:PlayerVO;
private var _isVictor:Boolean;
private var _rating:int;

private var _playerPortrait:ImageComponent;

private var _playerPortraitFrame:Bitmap;
private var _shipsInFleetBg:Bitmap;
private var _fleetFrame:Bitmap;

private var _participantBG:ScaleBitmap;

private var _playerName:Label;
private var _playerRating:Label;

private var _entityText:Vector.<Label>;

private var _ships:Vector.<ShipIcon>;

private var _getUIAsset:Function;
private var _loadPortraitMedium:Function;
private var _loadPortraitIcon:Function;

private var WINNER_FRAME_X:Number = 1;
private var LOSER_FRAME_X:Number = 16;

private var WINNER_PORTRAIT_X:Number = 5;
private var LOSER_PORTRAIT_X:Number = 330;

private var WINNER_BATTLE_TEXT_X:Number = 135;
private var LOSER_BATTLE_TEXT_X:Number = 118;

private var BATTLE_ENTITY_Y:Number = 58;

private var WINNER_BATTLE_SHIP_X:Number = 300;
private var LOSER_BATTLE_SHIP_X:Number = 25;
private var BATTLE_SHIP_Y:Number = 8;

private var _leftEntityText:String = 'CodeString.BattleLogs.EntityHealthLeft';
private
```

```
var _RightEntityText:String = 'CodeString.BattleLogs.EntityHealthRight';
private var _fleetRatingText:String = 'CodeString.BattleLogs.FleetRating';
private var _baseRatingText:String = 'CodeString.BattleLogs.BaseRating';
private var _baseHealthText:String = 'CodeString.BattleLog.BaseHealth';

public function BattleEndParticipantDisplay( player:PlayerVO, rating:int, isVictor:Boolean )
{
super();
var factionColor:uint = CommonFunctionUtil.getFactionColor(player.faction);

_player = player;
_isVictor = isVictor;
_rating = rating;

_participantBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_participantBG.width = 449;
_participantBG.height = 164;

_fleetFrame = UIFactory.getBitmap('SectorFleetSelectionBGBMD');

_playerPortrait = ObjectPool.get(ImageComponent);
_playerPortrait.init(2000, 2000);

_playerPortraitFrame = UIFactory.getScaleBitmap(PanelEnum.CHARACTER_FRAME);
_playerPortraitFrame.width = 130;
_playerPortraitFrame.height = 130;

_playerName = new Label(24, 0xf0f0f0, 210, 25);
_playerName.constrictTextToSize = false;
_playerName.align = (_isVictor) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
_playerName.text = player.name;
_playerName.textColor = factionColor;

_playerRating = new Label(20, 0xfbefaf, 210, 25);
_playerRating.constrictTextToSize = false;
_playerRating.align = (_isVictor) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
_playerRating.constrictTextToSize = false;

_ships = new Vector.<ShipIcon>;
_entityText = new Vector.<Label>;

addChild(_participantBG);
addChild(_fleetFrame);
addChild(_playerPortrait);
addChild(_playerName);
addChild(_playerRating);
addChild(_playerPortraitFrame);
}

public function setUp( getUIAsset:Function, loadPortraitMedium:Function,
loadPortraitIcon:Function,
```

```
getBattleEntities:Function ):void
{
_getUIAsset = getUIAsset;
_loadPortraitMedium = loadPortraitMedium;
_loadPortraitIcon = loadPortraitIcon;

var battleEntities:Vector.<BattleEntityVO> = getBattleEntities(_player.id);

var ratingText:String = (battleEntities.length > 0) ? _fleetRatingText : _baseRatingText;

_playerRating.setTextWithTokens(ratingText, {'[[Number.Rating]]':_rating});

if (battleEntities.length > 0)
setUpPlayerFleet(battleEntities);
else
setUpBase(getBattleEntities(_player.id, CategoryEnum.BUILDING));

_loadPortraitMedium(_player.avatarName, _playerPortrait.onImageLoaded);
layout();
}

private function layout():void
{
_participantBG.x = (_isVictor) ? WINNER_FRAME_X : LOSER_FRAME_X;

_playerName.x = (_isVictor) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
_playerName.y = 8;

_playerRating.x = (_isVictor) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
_playerRating.y = 35;

_playerPortrait.x = ((_isVictor) ? WINNER_PORTRAIT_X : LOSER_PORTRAIT_X) + 5;
_playerPortrait.y = 22;

_playerPortraitFrame.x = ((_isVictor) ? WINNER_PORTRAIT_X : LOSER_PORTRAIT_X);
_playerPortraitFrame.y = 17;

_fleetFrame.x = (_isVictor) ? WINNER_BATTLE_SHIP_X - 1 : LOSER_BATTLE_SHIP_X - 1;
_fleetFrame.y = BATTLE_SHIP_Y - 1;

var xPos:Number = (_isVictor) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
var yPos:Number = BATTLE_ENTITY_Y;
var len:uint = _entityText.length;
var currentLabel:Label;
for (var i:uint = 0; i < len; ++i)
{
currentLabel = _entityText[i];
currentLabel.x = xPos;
currentLabel.y = yPos;

yPos
```

```
+= currentLabel.textHeight;
}
}

private function setUpBase( buildings:Vector.<BattleEntityVO> ):void
{
_fleetFrame.visible = false;
var len:uint = buildings.length;
var baseHealth:Number = 0;
var allBldgsCurrentHealth:Number = 0;
var allBldgsMaxHealth:Number = 0;
var currentBuilding:BattleEntityVO;
var currentBuildingProto:IPrototype;
var currentBuildingMaxHealth:Number;
for (var i:uint = 0; i < len; ++i)
{
currentBuilding = buildings[i];
currentBuildingProto = currentBuilding.prototype;
if (currentBuildingProto)
{
if(currentBuildingProto.itemClass != TypeEnum.PYLON)
{
currentBuildingMaxHealth = Number(currentBuildingProto.getValue('health'));
allBldgsCurrentHealth += currentBuilding.healthPercent * currentBuildingMaxHealth;
allBldgsMaxHealth += currentBuildingMaxHealth;
}
}
}

baseHealth = allBldgsMaxHealth > 0 ? allBldgsCurrentHealth / allBldgsMaxHealth : 1.0;

var healthValue:Number = Math.round(baseHealth * 100);
var healthColor:uint = getHealthColor(healthValue);
var textToUse:String = (_isVictor) ? _leftEntityText : _RightEntityText;
var base:Label = new Label(12, 0xf0f0f0, 210, 25);
base.align = (_isVictor) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
base.constrictTextToSize = false;
base.x = (_isVictor) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
base.y = BATTLE_ENTITY_Y;
base.setHtmlTextWithTokens(textToUse, {'[[HexNumber.Color]]':healthColor.toString(16),
'[[Number.EntityHealth]]':healthValue, '[[String.EntityName]]':_baseHealthText});
_entityText.push(base);
addChild(base);
}

private function setUpPlayerFleet( ships:Vector.<BattleEntityVO> ):void
{
_fleetFrame.visible = true;

var image:ShipIcon;
var
```

```
xPos:Number;
var yPos:Number;

var shipProto:IPrototype;
var currentBattleEntity:BattleEntityVO;
var len:uint = ships.length;
for (var i:uint = 0; i < 6; ++i)
{
if (i < len)
currentBattleEntity = ships[i];

image = new ShipIcon();
image.scale(0.37, 0.37);

xPos = (_isVictor) ? WINNER_BATTLE_SHIP_X : LOSER_BATTLE_SHIP_X;
yPos = BATTLE_SHIP_Y;

switch (i)
{
case 0:
xPos += 46;
yPos += 0;
break;
case 1:
xPos += 0;
yPos += 25;
break;
case 2:
xPos += 90;
yPos += 25;
break;
case 3:
xPos += 0;
yPos += 76;
break;
case 4:
xPos += 90;
yPos += 76;
break;
case 5:
xPos += 45;
yPos += 104;
break;
}

if (currentBattleEntity)
{
shipProto = currentBattleEntity.prototype;
if (shipProto)
{
image.onLoadShipImage.add(_loadPortraitIcon);
```

```
image.setShip(null, shipProto);
image.setBarValue(1 - currentBattleEntity.healthPercent);
addShipLabel(shipProto, currentBattleEntity.healthPercent);
}
currentBattleEntity = null;
}

image.x = xPos;
image.y = yPos;
image.mouseEnabled = false;
addChild(image);
_ships.push(image);
}

}

private function addShipLabel( shipProto:IPrototype, health:Number ):void
{
var healthValue:Number = Math.round(health * 100);
var shipUIAsset:AssetVO = _getUIAsset(shipProto);
var healthColor:uint = getHealthColor(healthValue);
var textToUse:String = (_isVictor) ? _leftEntityText : _RightEntityText;
var ship:Label = new Label(14, 0xf0f0f0, 210, 25);
ship.constrictTextToSize = false;
ship.align = (_isVictor) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
ship.setHtmlTextWithTokens(textToUse, {'[[HexNumber.Color]]':healthColor.toString(16),
'[[Number.EntityHealth]]':healthValue, '[[String.EntityName]]':shipUIAsset.visibleName});
_entityText.push(ship);
addChild(ship);

}

private function getHealthColor( healthValue:Number ):uint
{

if (healthValue >= 75)
return 0x3cf219;
else if (healthValue >= 50)
return 0xfbe81a;
else if (healthValue >= 25)
return 0xfa7d0e;
else
return 0xf81919;
}

override public function get height():Number
{
return _participantBG.height;
}

public
```

```
function destroy():void
{
_participantBG = null;

ObjectPool.give(_playerPortrait);
_playerPortrait = null;

_playerName.destroy();
_playerName = null;

_playerRating.destroy();
_playerRating = null;

_playerPortraitFrame = null;

var len:uint = _ships.length;
var i:uint;
var currentShipIcon:ShipIcon;
for (; i < len; ++i)
{
currentShipIcon = _ships[i];
currentShipIcon.destroy();
currentShipIcon = null;
}
_ships.length = 0;
len = _entityText.length;
var currentLabel:Label;
for (i = 0; i < len; ++i)
{
currentLabel = _entityText[i];
currentLabel.destroy();
currentLabel = null;
}

_isVictor = false;

_getUIAsset = null;
_loadPortraitMedium = null;
_loadPortraitIcon = null;
}
}
}

------------------------------------
File 788: igw\com\ui\modal\battle\BattleEndView.as
package com.ui.modal.battle
{
import com.Application;
import com.enum.AudioEnum;
import com.enum.server.UnavailableRerollEnum;
import
```

```
com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.presenter.battle.IBattlePresenter;
import com.service.ExternalInterfaceAPI;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.battle.chance.ChanceGameDisplayComponent;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

import org.adobe.utils.DictionaryUtil;
import org.adobe.utils.StringUtil;
import org.shared.ObjectPool;

public class BattleEndView extends View
{

[Inject]
private var _tooltips:Tooltips;

private var _battleID:String;
private var _blueprintProtoName:String;

private var _victors:Dictionary;

private var _blueprintCost:int;
private var _lootedAlloyAmount:Number;
private var _lootedCreditsAmount:Number;
private var _lootedEnergyAmount:Number;
private
```

```
var _lootedSyntheticsAmount:Number;

private var _cargoFull:Boolean;

private var _maxHeight:Number;

private var _participants:Vector.<String>;

private var _isPlayerInCombat:Boolean;
private var _isPlayerVictor:Boolean;

private var _bg:DefaultWindowBG;

private var _okBtn:BitmapButton;
private var _shareButton:BitmapButton;

private var _defeatedVO:PlayerVO;
private var _victorVO:PlayerVO;

private var _lootHolder:Sprite;
private var _participantHolder:Sprite;

private var _scrollbar:VScrollbar;

private var _scrollRect:Rectangle;

private var _lootedAlloySymbol:Bitmap;
private var _lootedCreditsSymbol:Bitmap;
private var _lootedEnergySymbol:Bitmap;
private var _lootedSyntheticsSymbol:Bitmap;

private var _headingBG:ScaleBitmap;
private var _participantsBG:ScaleBitmap;
private var _beCDialogueBG:ScaleBitmap;
private var _lootBg:ScaleBitmap;
private var _unavailableRerollBG:ScaleBitmap;

private var _titleLbl:Label;
private var _lootedAlloyLbl:Label;
private var _lootedCreditsLbl:Label;
private var _lootedEnergyLbl:Label;
private var _lootedSyntheticsLbl:Label;
private var _beCDialogueLbl:Label;
private var _victorsHeading:Label;
private var _losersHeading:Label;
private var _unavailableReroll:Label;

private var _blueprint:BlueprintVO;

private var _chanceComponent:ChanceGameDisplayComponent;

private
```

```
var _soundToPlay:String = '';
private var _cargoFullString:String = 'CodeString.BattleEnd.Full'; //[[Number.AmountLooted]] (FULL)
private var _okBtnText:String = 'CodeString.Shared.OkBtn'; //OK
private var _shareBtnText:String = 'CodeString.BattleEnd.ShareBtn'; //Share
private var _completeBtnText:String = 'CodeString.Dialogue.Complete'; //COMPLETE
private var _drawText:String = 'CodeString.BattleLog.DrawTitle' //DRAW
private var _watchText:String = 'CodeString.BattleEnd.Watch'; //WATCH
private var _victoryText:String = 'CodeString.BattleEnd.Victory'; //VICTORY
private var _defeatText:String = 'CodeString.BattleEnd.Defeat'; //DEFEAT
private var _titleText:String = 'CodeString.BattleEnd.Title'; //BATTLE REPORT
private var _victorText:String = 'CodeString.BattleLog.Victor'; //Victor
private var _defeatedText:String = 'CodeString.BattleLog.Defeated'; //Defeated

private var _NoChallenge:String = 'CodeString.UnavailableReroll.NoChallenge'; //Your fleet's rating was too high to receive blueprints from this opponent.
private var _FarmPenalty:String = 'CodeString.UnavailableReroll.FarmPenalty'; //You have attacked this opponent too often to receive blueprints.
private var _DamageReq:String = 'CodeString.UnavailableReroll.DamageReq'; //You did not do enough damage to this opponent to receive a blueprint.
private var _AllComplete:String = 'CodeString.UnavailableReroll.AllComplete'; //You have already completed all blueprints which can drop from this opponent.
private var _SlotsFull:String = 'CodeString.UnavailableReroll.SlotsFull'; //This opponent was not eligible to drop parts for any of your in-progress blueprints.
private var _NoneInBand:String = 'CodeString.UnavailableReroll.NoneInBand'; //Opponents of this rating have no blueprints available for you.
private var _MissingTech:String = 'CodeString.UnavailableReroll.MissingTech'; //You must upgrade your labs or technology to receive blueprints from this opponent.
private var _BadEncounter:String = 'CodeString.UnavailableReroll.BadEncounter'; //The specific opponent you killed could not drop any blueprints for you.
private var _BadFaction:String = 'CodeString.UnavailableReroll.BadFaction'; //You must fight opponents of a different faction to receive blueprints.
private var _NoneAtRarity:String = 'CodeString.UnavailableReroll.NoneAtRarity'; //There are no other blueprints available of the specified rarity.

[PostConstruct]
override public function init():void
{
super.init();
presenter.addCleanupListener(destroy);

var totalVictors:uint = DictionaryUtil.getLength(_victors);

_isPlayerInCombat = presenter.isPlayerInCombat(CurrentUser.id);
_isPlayerVictor = (_isPlayerInCombat) ? (CurrentUser.id in _victors) : false;
_participants = presenter.participants;
_maxHeight = 0;

if (_isPlayerVictor)
_victorVO = CurrentUser.user;

var
```

```
greaterThenTwoParticipants:Boolean = (_participants.length > 2);

var soundNumber:int = Math.random() * (4 - 1) + 1;
var sound:String;
if (_isPlayerVictor || !_isPlayerInCombat)
{

switch (soundNumber)
{
case 1:
sound = AudioEnum.AFX_STG_VICTORY_1;
break;
case 2:
sound = AudioEnum.AFX_STG_VICTORY_2;
break;
case 3:
sound = AudioEnum.AFX_STG_VICTORY_3;
break;
case 4:
sound = AudioEnum.AFX_STG_VICTORY_4;
break;
}

} else
{
switch (soundNumber)
{
case 1:
sound = AudioEnum.AFX_STG_DEFEAT_1;
break;
case 2:
sound = AudioEnum.AFX_STG_DEFEAT_2;
break;
case 3:
sound = AudioEnum.AFX_STG_DEFEAT_3;
break;
case 4:
sound = AudioEnum.AFX_STG_DEFEAT_4;
break;
}

}
presenter.playSound(sound, 0.25);

var bgWidth:Number = (greaterThenTwoParticipants) ? 946 : 926;

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(bgWidth, 479);
_bg.addTitle(_titleText, 239);
_bg.x -= 21;
addListener(_bg.closeButton,
```

```
MouseEvent.CLICK, exitCombat);
addChild(_bg);

_chanceComponent = new ChanceGameDisplayComponent();
_chanceComponent.onPurchaseComplete.add(onBlueprintPurchase);
_chanceComponent.rerollClickSignal.add(onRerollClick);
_chanceComponent.scanClickSignal.add(onScanClick);
_chanceComponent.denyClickSignal.add(onDenyClick);
_chanceComponent.getBlueprintPrototype = presenter.getBlueprintPrototypeByName;
_chanceComponent.getResearchPrototypeByName =
presenter.getResearchPrototypeByName;
_chanceComponent.getSelectionInfo = getSelectionInfo;
_chanceComponent.cargoFull = _cargoFull;
_chanceComponent.componentHolder.x = (greaterThenTwoParticipants) ? 460 : 440;
_chanceComponent.componentHolder.y = 324;

_titleLbl = new Label(80, 0xffdd3d, 914, 100);
_titleLbl.allCaps = true;
_titleLbl.constrictTextToSize = false;
_titleLbl.multiline = false;
_titleLbl.align = TextFormatAlign.CENTER;
_titleLbl.letterSpacing = 1.5;
_titleLbl.y = 54;

if (!_isPlayerInCombat)
_titleLbl.text = _watchText;
else if (_isPlayerVictor)
_titleLbl.text = _victoryText;
else if (totalVictors == 0)
_titleLbl.text = _drawText;
else
{
_titleLbl.textColor = 0xF58993;
_titleLbl.text = _defeatText;
}

if (totalVictors > 0)
{
_victorsHeading = new Label(22, 0xffdd3d, 130, 33);
_victorsHeading.x = 11;
_victorsHeading.y = 125;
_victorsHeading.align = TextFormatAlign.CENTER;
_victorsHeading.text = _victorText;

_losersHeading = new Label(22, 0xfe4f4f, 130, 33);
_losersHeading.x = 769;
_losersHeading.y = 125;
_losersHeading.align = TextFormatAlign.CENTER;
_losersHeading.text = _defeatedText;
}

_lootBg
```

```
= UIFactory.getScaleBitmap(PanelEnum.CONTAINER_NOTCHED);
_lootBg.width = 362;
_lootBg.height = 123;
_lootBg.x = 7;
_lootBg.y = 324;

_lootHolder = new Sprite();
_lootHolder.x = 13;
_lootHolder.y = 335;

_lootedCreditsSymbol = UIFactory.getBitmap('LootedResourceCreditsBMD');

_lootedEnergySymbol = UIFactory.getBitmap('LootedResourceEnergyBMD');
_lootedEnergySymbol.x = _lootedCreditsSymbol.x + 179;
_lootedEnergySymbol.y = _lootedCreditsSymbol.y;

_lootedAlloySymbol = UIFactory.getBitmap('LootedResourceAlloyBMD');
_lootedAlloySymbol.x = _lootedCreditsSymbol.x;
_lootedAlloySymbol.y = _lootedCreditsSymbol.y + 50;

_lootedSyntheticsSymbol = UIFactory.getBitmap('LootedResourceSyntheticsBMD');
_lootedSyntheticsSymbol.x = _lootedEnergySymbol.x;
_lootedSyntheticsSymbol.y = _lootedCreditsSymbol.y + 50;

var textColor:uint = (_isPlayerVictor) ? 0x7afe60 : 0xf04c4c;

_lootedCreditsLbl = new Label(13, 0xf0f0f0, 140, 20, true, 1);
_lootedCreditsLbl.textColor = textColor;
_lootedCreditsLbl.x = _lootedCreditsSymbol.width * 0.5 + _lootedCreditsSymbol.x - 35;
_lootedCreditsLbl.y = _lootedCreditsSymbol.height * 0.5 + _lootedCreditsSymbol.y - 4;
_lootedCreditsLbl.constrictTextToSize = false;
_lootedCreditsLbl.align = TextFormatAlign.LEFT;
_lootedCreditsLbl.letterSpacing = 1.5;

_lootedEnergyLbl = new Label(13, 0xf0f0f0, 140, 30, true, 1);
_lootedEnergyLbl.textColor = textColor;
_lootedEnergyLbl.x = _lootedEnergySymbol.width * 0.5 + _lootedEnergySymbol.x - 35;
_lootedEnergyLbl.y = _lootedEnergySymbol.height * 0.5 + _lootedEnergySymbol.y - 4;
_lootedEnergyLbl.constrictTextToSize = false;
_lootedEnergyLbl.align = TextFormatAlign.LEFT;
_lootedEnergyLbl.letterSpacing = 1.5;


_lootedAlloyLbl = new Label(13, 0xf0f0f0, 140, 30, true, 1);
_lootedAlloyLbl.textColor = textColor;
_lootedAlloyLbl.x = _lootedAlloySymbol.width * 0.5 + _lootedAlloySymbol.x - 35;
_lootedAlloyLbl.y = _lootedAlloySymbol.height * 0.5 + _lootedAlloySymbol.y - 4;
_lootedAlloyLbl.constrictTextToSize = false;
_lootedAlloyLbl.align = TextFormatAlign.LEFT;
_lootedAlloyLbl.letterSpacing = 1.5;

_lootedSyntheticsLbl
```

```
= new Label(13, 0xf0f0f0, 140, 30, true, 1);
_lootedSyntheticsLbl.textColor = textColor;
_lootedSyntheticsLbl.x = _lootedSyntheticsSymbol.width * 0.5 + _lootedSyntheticsSymbol.x -
35;
_lootedSyntheticsLbl.y = _lootedSyntheticsSymbol.height * 0.5 + _lootedSyntheticsSymbol.y - 4;
_lootedSyntheticsLbl.constrictTextToSize = false;
_lootedSyntheticsLbl.align = TextFormatAlign.LEFT;
_lootedSyntheticsLbl.letterSpacing = 1.5;

_lootedCreditsLbl.text = StringUtil.commaFormatNumber(_lootedCreditsAmount);

if (_isPlayerVictor && _cargoFull)
{
_lootedEnergyLbl.setTextWithTokens(_cargoFullString,
{'[[Number.AmountLooted]]':StringUtil.commaFormatNumber(_lootedEnergyAmount)});
_lootedAlloyLbl.setTextWithTokens(_cargoFullString,
{'[[Number.AmountLooted]]':StringUtil.commaFormatNumber(_lootedAlloyAmount)});
_lootedSyntheticsLbl.setTextWithTokens(_cargoFullString,
{'[[Number.AmountLooted]]':StringUtil.commaFormatNumber(_lootedSyntheticsAmount)});
} else
{
_lootedEnergyLbl.text = StringUtil.commaFormatNumber(_lootedEnergyAmount);
_lootedAlloyLbl.text = StringUtil.commaFormatNumber(_lootedAlloyAmount);
_lootedSyntheticsLbl.text = StringUtil.commaFormatNumber(_lootedSyntheticsAmount);
}

_beCDialogueLbl = new Label(21, 0x7ca5fd, 579, 60, true);
_beCDialogueLbl.align = TextFormatAlign.LEFT;

_okBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 140, 40, 760, 461, _okBtnText);
addListener(_okBtn, MouseEvent.CLICK, exitCombat);

var debugFB:Boolean = false;
if (debugFB ||
(Application.NETWORK == Application.NETWORK_FACEBOOK &&
!presenter.isPVEBattle &&
_isPlayerInCombat &&
_isPlayerVictor))
{
_shareButton = UIFactory.getButton(ButtonEnum.BLUE_A, 140, 40, 600, 461, _shareBtnText);
addListener(_shareButton, MouseEvent.CLICK, onClickShare);
}

_headingBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_headingBG.width = (greaterThenTwoParticipants) ? 918 : 898;
_headingBG.height = 102;
_headingBG.x = 7;
_headingBG.y = 48;

_participantsBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_participantsBG.width
```

```
= 898;
_participantsBG.height = 164;
_participantsBG.x = 7;
_participantsBG.y = 154;

_beCDialogueBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_beCDialogueBG.width = 582;
_beCDialogueBG.height = 37;
_beCDialogueBG.x = 7;
_beCDialogueBG.y = 465;

var participants:Vector.<PlayerVO> = new Vector.<PlayerVO>;
var len:uint = _participants.length;
var i:uint;
for (; i < len; ++i)
{
participants.push(presenter.getPlayer(_participants[i]));
}

_participantHolder = new Sprite();
_participantHolder.x = 6;
_participantHolder.y = 154;

var currentBattlEndParticipantDisplay:BattleEndParticipantDisplay;
var currentPlayer:PlayerVO;
var isVictor:Boolean;
var victorCount:uint;
var defeatedCount:uint;
len = participants.length;
for (i = 0; i < len; ++i)
{
currentPlayer = participants[i];
isVictor = (totalVictors > 0) ? currentPlayer.id in _victors : (i % 2 == 0);
currentBattlEndParticipantDisplay = new BattleEndParticipantDisplay(currentPlayer,
presenter.getParticipantRating(currentPlayer.id), isVictor);
currentBattlEndParticipantDisplay.setUp(presenter.getAssetVO, presenter.loadMediumImage,
presenter.loadMiniIconFromEntityData, presenter.getBattleEntitiesByPlayer);
currentBattlEndParticipantDisplay.x = (isVictor) ? 6 : 440;
currentBattlEndParticipantDisplay.y = (isVictor) ? ((victorCount++ *
(currentBattlEndParticipantDisplay.height + 14))) : ((defeatedCount++ *
(currentBattlEndParticipantDisplay.
height + 14)));
_participantHolder.addChild(currentBattlEndParticipantDisplay);

if (_maxHeight < currentBattlEndParticipantDisplay.y + currentBattlEndParticipantDisplay.height)
_maxHeight = (currentBattlEndParticipantDisplay.y + currentBattlEndParticipantDisplay.height)

if (_defeatedVO == null && !isVictor)
_defeatedVO = currentPlayer;

if
```

```
(_victorVO == null && isVictor)
_victorVO = currentPlayer;
}

_scrollRect = new Rectangle(_participantHolder.x, _participantHolder.y, _participantsBG.width,
_participantsBG.height);
_scrollRect.y = 0;
_participantHolder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _participantsBG.x + _participantsBG.width + 5;
var scrollbarYPos:Number = _participantsBG.y;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 123;

_lootHolder.addChild(_lootedCreditsSymbol);
_lootHolder.addChild(_lootedAlloySymbol);
_lootHolder.addChild(_lootedSyntheticsSymbol);
_lootHolder.addChild(_lootedEnergySymbol);
_lootHolder.addChild(_lootedCreditsLbl);
_lootHolder.addChild(_lootedAlloyLbl);
_lootHolder.addChild(_lootedSyntheticsLbl);
_lootHolder.addChild(_lootedEnergyLbl);

addChild(_bg);
addChild(_headingBG);
addChild(_participantsBG);
addChild(_beCDialogueBG);
addChild(_titleLbl);
addChild(_lootBg);
addChild(_lootHolder);
addChild(_beCDialogueLbl);
addChild(_okBtn);
addChild(_participantHolder);
addChild(_scrollbar);

showChanceComponent();

if (_victorsHeading)
addChild(_victorsHeading);

if (_losersHeading)
addChild(_losersHeading);

if (_shareButton)
addChild(_shareButton);
```

```
addBattleEndDialogue();

presenter.addRerollFromRerollCallback(onRerollUpdated);
presenter.addRerollFromScanCallback(onRerollUpdated);

addEffects();
effectsIN();
}

private function showChanceComponent( brVO:BattleRerollVO = null ):void
{

if (!brVO)
_chanceComponent.battleRerollVO = presenter.getAvailableRerollById(_battleID);
else
_chanceComponent.battleRerollVO = brVO;

if (_blueprintProtoName)
{
var blueprintVO:IPrototype = presenter.getBlueprintPrototypeByName(_blueprintProtoName);
var bpAsset:AssetVO = presenter.getAssetVO(blueprintVO);
ExternalInterfaceAPI.shareBlueprintFind(blueprintVO);
_blueprint = presenter.getBlueprintByName(_blueprintProtoName);

_chanceComponent.loadIconSignal.add(loadBPIcon);
if (_blueprint)
_blueprintCost = presenter.getBlueprintHardCurrencyCost(_blueprint,
_blueprint.partsRemaining);

_chanceComponent.init(_blueprint);
_chanceComponent.rollCost = presenter.getConstantPrototypeValueByName('rerollItemPrice');

_chanceComponent.showBlueprint(_blueprintProtoName, null, bpAsset, _blueprintCost);

_tooltips.addTooltip(_chanceComponent.blueprintShipIcon, _chanceComponent,
_chanceComponent.getTooltip);

addChild(_chanceComponent);

} else if (_isPlayerVictor && _chanceComponent.battleRerollVO)
{
_chanceComponent.scanCost =
presenter.getConstantPrototypeValueByName('rerollLootPrice');
_chanceComponent.showScanView();
addChild(_chanceComponent);
} else
{
_unavailableRerollBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_NOTCHED);
_unavailableRerollBG.width
```

```
= 449;
_unavailableRerollBG.height = 123;
_unavailableRerollBG.x = (_participantHolder.numChildren > 2) ? 476 : 456;
_unavailableRerollBG.y = 324;

_unavailableReroll = new Label(24, 0xffdd3d, _unavailableRerollBG.width,
_unavailableRerollBG.height);
_unavailableReroll.align = TextFormatAlign.CENTER;
_unavailableReroll.multiline = true;
_unavailableReroll.text = getUnavailableRerollString(presenter.getUnavailableReroll(_battleID));
_unavailableReroll.x = _unavailableRerollBG.x;
_unavailableReroll.y = _unavailableRerollBG.y + (_unavailableRerollBG.height -
_unavailableReroll.textHeight) * 0.5;

addChild(_unavailableRerollBG);
addChild(_unavailableReroll);
}
}

public function onRerollUpdated( rerollVO:BattleRerollVO ):void
{
if (rerollVO.blueprintPrototype != '')
{
_blueprintProtoName = rerollVO.blueprintPrototype;
_chanceComponent.hideBlueprint();
showChanceComponent(rerollVO);
} else
_chanceComponent.showGainedResourcesView(rerollVO);
}

private function onChangedScroll( percent:Number ):void
{
if (_scrollRect)
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;

if (_participantHolder)
_participantHolder.scrollRect = _scrollRect;
}

private function loadBPIcon( imageName:String, callback:Function ):void
{
if (presenter)
presenter.loadIcon(imageName, callback);
}

protected function exitCombat( e:MouseEvent ):void
{
presenter.exitCombat();
destroy();
}

private
```

```
function addBattleEndDialogue():void
{
var dialogueOptions:Vector.<IPrototype> = new Vector.<IPrototype>;
if (_isPlayerInCombat)
{
if (_isPlayerVictor && _victorVO)
dialogueOptions = presenter.getBEDialogueByFaction(_victorVO.faction);
else
{
if (_victorVO)
dialogueOptions = presenter.getBEDialogueByFaction(_victorVO.faction, 'Taunt');
else
{
if (_defeatedVO)
dialogueOptions = presenter.getBEDialogueByFaction(_defeatedVO.faction, 'Taunt');
}
}

} else
{
if (_victorVO && CurrentUser)
dialogueOptions = presenter.getBEDialogueByFaction(_victorVO.faction, (_victorVO.faction ==
CurrentUser.faction) ? 'Victory' : 'Taunt');
else
{
if (_defeatedVO)
dialogueOptions = presenter.getBEDialogueByFaction(_defeatedVO.faction, 'Taunt');
}
}

if(dialogueOptions.length>0)
{
_beCDialogueLbl.text = dialogueOptions[Math.floor(CommonFunctionUtil.randomMinMax(0,
(dialogueOptions.length - 1)))].getValue('dialogString');

}
else
{
_beCDialogueLbl.text = "";
_soundToPlay = "";
}

_beCDialogueLbl.x = _beCDialogueBG.x + (_beCDialogueBG.width -
_beCDialogueLbl.textWidth) * 0.5;
_beCDialogueLbl.y = _beCDialogueBG.y + (_beCDialogueBG.height -
_beCDialogueLbl.textHeight) * 0.5;
}

private function getUnavailableRerollString( v:int ):String
{
switch
```

```
(v)
{
case UnavailableRerollEnum.NO_CHALLENGE:
return _NoChallenge;
case UnavailableRerollEnum.FARM_PENALTY:
return _FarmPenalty;
case UnavailableRerollEnum.DAMAGE_REQ:
return _DamageReq;
case UnavailableRerollEnum.ALL_COMPLETE:
return _AllComplete;
case UnavailableRerollEnum.SLOTS_FULL:
return _SlotsFull;
case UnavailableRerollEnum.NONE_IN_BAND:
return _NoneInBand;
case UnavailableRerollEnum.MISSING_TECH:
return _MissingTech;
case UnavailableRerollEnum.BAD_ENCOUNTER:
return _BadEncounter;
case UnavailableRerollEnum.BAD_FACTION:
return _BadFaction;
case UnavailableRerollEnum.NONE_AT_RARITY:
return _NoneAtRarity;

}

return '';
}

private function getSelectionInfo( prototype:IPrototype, info:Function ):*
{
var selectionInfo:*;
if (prototype)
{
var reqBuildClass:String = prototype.getUnsafeValue('requiredBuildingClass');
if (reqBuildClass)
{
var proto:IPrototype;
if (reqBuildClass == 'ShipDesignFacility')
{
proto = presenter.getShipPrototype(prototype.getValue('referenceName'));
if (proto)
{
selectionInfo = info(proto.getValue('type'), proto);
} else
{
proto = presenter.getShipPrototype(prototype.getValue('referenceName'));
if (proto)
selectionInfo = info(proto.getValue('type'), proto);
}

}
```

```
else if (reqBuildClass == 'CommandCenter')
{
selectionInfo = info(prototype.getValue('type'), prototype);
} else if (reqBuildClass == 'WeaponsDesignFacility')
{
proto = presenter.getModulePrototypeByName(prototype.getValue('referenceName'));
selectionInfo = info(proto.getValue('type'), proto);
} else
{
proto = presenter.getModulePrototypeByName(prototype.getValue('referenceName'));
selectionInfo = info(proto.getValue('type'), proto);
}
} else
selectionInfo = info(prototype.getValue('type'), prototype);
}

return selectionInfo;
}

private function onClickShare( e:MouseEvent ):void
{
presenter.sharePvPVictory();
}

private function onBlueprintPurchase( blueprintID:String, battleID:String ):void
{
var vo:BlueprintVO = presenter.getBlueprintByID(blueprintID);
presenter.purchaseBlueprint(vo, vo.partsRemaining);
presenter.removeRerollFromAvailable(battleID);
}

private function onRerollClick( battleID:String, name:String ):void
{
presenter.removeBlueprintByName(name);
presenter.purchaseReroll(battleID);
}

private function onScanClick( battleID:String ):void
{
presenter.purchaseDeepScan(battleID);
}

private function onDenyClick( battleID:String ):void
{
presenter.removeRerollFromAvailable(battleID);
}

public function get lootHolder():Sprite { return _lootHolder; }
public function set battleID( v:String ):void { _battleID = v; }
public function set victors( v:Dictionary ):void { _victors = v; }
public
```

```
function set lootedAlloyAmount( v:Number ):void { _lootedAlloyAmount = v; }
public function set lootedCreditsAmount( v:Number ):void { _lootedCreditsAmount = v; }
public function set lootedEnergyAmount( v:Number ):void { _lootedEnergyAmount = v; }
public function set lootedSyntheticsAmount( v:Number ):void { _lootedSyntheticsAmount = v; }
public function set cargoFull( v:Boolean ):void { _cargoFull = v; }
public function set blueprintProtoName( v:String ):void { _blueprintProtoName = v; }

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IBattlePresenter ):void { _presenter = value; }
public function get presenter():IBattlePresenter { return IBattlePresenter(_presenter); }
[Inject]
public function set tooltips( value:Tooltips ):void { _tooltips = value; }

override public function destroy():void
{
presenter.removeCleanupListener(destroy);
presenter.removeRerollFromRerollCallback(onRerollUpdated);
presenter.removeRerollFromScanCallback(onRerollUpdated);
super.destroy();

if (_tooltips)
_tooltips.removeTooltip(null, this);

_tooltips = null;

if (_bg)
{
removeListener(_bg.closeButton, MouseEvent.CLICK, exitCombat);
ObjectPool.give(_bg);
}

_bg = null;

if (_okBtn)
{
removeListener(_okBtn, MouseEvent.CLICK, exitCombat);
_okBtn.destroy();
}

_okBtn = null;


if (_shareButton)
{
removeListener(_shareButton, MouseEvent.CLICK, onClickShare);
_shareButton.destroy();
}

_shareButton
```

```
= null;

_defeatedVO = null;
_victorVO = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar = null;

if (_titleLbl)
_titleLbl.destroy();

_titleLbl = null;

if (_lootedAlloyLbl)
_lootedAlloyLbl.destroy();

_lootedAlloyLbl = null;

if (_lootedCreditsLbl)
_lootedCreditsLbl.destroy();

_lootedCreditsLbl = null;

if (_lootedEnergyLbl)
_lootedEnergyLbl.destroy();

_lootedEnergyLbl = null;

if (_lootedSyntheticsLbl)
_lootedSyntheticsLbl.destroy();

_lootedSyntheticsLbl = null;

if (_beCDialogueLbl)
_beCDialogueLbl.destroy();

_beCDialogueLbl = null;

if (_victorsHeading)
_victorsHeading.destroy();

_victorsHeading = null;

if (_losersHeading)
_losersHeading.destroy();

_losersHeading = null;

if
```

```
(_unavailableReroll)
_unavailableReroll.destroy();

_unavailableReroll = null;

if (_chanceComponent)
_chanceComponent.destroy();

_chanceComponent = null;

var len:uint = _participantHolder.numChildren;
var currentParticipant:BattleEndParticipantDisplay;
for (var i:uint = 0; i < len; ++i)
{
currentParticipant = BattleEndParticipantDisplay(_participantHolder.getChildAt(i));
currentParticipant.destroy();
}
_participantHolder = null;

_lootHolder = null;
_scrollRect = null;

_lootedAlloySymbol = null;
_lootedCreditsSymbol = null;
_lootedEnergySymbol = null;
_lootedSyntheticsSymbol = null;

_headingBG = null;
_participantsBG = null;
_beCDialogueBG = null;
_lootBg = null;
_unavailableRerollBG = null;
}
}
}

-------------------------------------
File 789: igw\com\ui\modal\battle\BattleStartView.as
package com.ui.modal.battle
{
import com.presenter.battle.IBattlePresenter;
import com.ui.core.View;
import com.ui.core.component.label.Label;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.events.TimerEvent;
import flash.text.TextFormatAlign;
import flash.utils.Timer;

public
```

```
class BattleStartView extends View
{
private var _bg:Bitmap;
private var _endTick:int;
private var _seconds:int;
private var _startLabel:Label;
private var _timerLabel:Label;
private var _startTick:int;
private var _timer:Timer;

private var _startingBattle:String = 'CodeString.BattleBegin.Count'; //Starting battle in:

[PostConstruct]
override public function init():void
{
super.init();
presenter.addCleanupListener(destroy);
presenter.addStartListener(destroy);


_bg = PanelFactory.getPanel("CombatCountdownBGBMD");
addChild(_bg);

_startLabel = new Label(20, 0xf0f0f0, _bg.width, 100);
_startLabel.text = _startingBattle;
_startLabel.textColor = 0xf0f0f0;
_startLabel.align = TextFormatAlign.CENTER;
_startLabel.x = 0;
_startLabel.y = 45;
addChild(_startLabel);

_timerLabel = new Label(30, 0xf0f0f0, 100, 40);
_timerLabel.text = "00:00:00";
_timerLabel.textColor = 0xf0f0f0;
_timerLabel.x = 107;
_timerLabel.y = 107;
addChild(_timerLabel);

onTimer(null);
_timer = new Timer(1000);
_timer.addEventListener(TimerEvent.TIMER, onTimer);
_timer.start();

addEffects();
effectsIN();
}

public function update( start:int, end:int ):void
{
_startTick = start;
_endTick
```

```
= end;
_seconds = (_endTick - _startTick) * .1 + 1;
if (_timer)
{
onTimer(null);
_timer.reset();
_timer.start();
}
}

private function onTimer( e:TimerEvent ):void
{
_seconds--;

if (_seconds < 0)
_seconds = 0;

var secondsString:String = "00:00:";

if (_seconds.toString().length > 1)
secondsString += _seconds.toString();

else
secondsString += "0" + _seconds.toString();

_timerLabel.text = secondsString;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IBattlePresenter ):void { _presenter = value; }
public function get presenter():IBattlePresenter { return IBattlePresenter(_presenter); }

override public function get typeUnique():Boolean { return false; }

override public function destroy():void
{
presenter.removeCleanupListener(destroy);
presenter.removeStartListener(destroy);
super.destroy();
_bg = null;
_startLabel = null;
_timer.removeEventListener(TimerEvent.TIMER, onTimer);
_timer.stop();
_timer = null;
}
}
}

-----------------------------------
```

File 790: igw\com\ui\modal\battle\chance\ChanceGameDisplayComponent.as

```
package com.ui.modal.battle.chance
{
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.component.IComponent;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.filters.ColorMatrixFilter;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Timer;
import flash.utils.getDefinitionByName;

import org.adobe.utils.StringUtil;
import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class ChanceGameDisplayComponent extends Sprite implements IComponent
{
public var onPurchaseComplete:Signal;
public var loadIconSignal:Signal;
public var rerollClickSignal:Signal;
public var scanClickSignal:Signal;
public var denyClickSignal:Signal;

public var getBlueprintPrototype:Function;
public
```

```
var getResearchPrototypeByName:Function;
public var getSelectionInfo:Function;

private var _componentHolder:Sprite;

private var _cargoFull:Boolean;
private var _blueprintFrame:Bitmap;
private var _blueprintShipIcon:ImageComponent;

private var _purchaseBlueprintBtn:BitmapButton;
private var _rerollBtn:BitmapButton;
private var _deepScanBtn:BitmapButton;
private var _denyBtn:BitmapButton;

private var _blueprintBG:Bitmap;
private var _blueprintNameBG:Bitmap;
private var _blueprintGlow:Bitmap;
private var _blueprintPremiumSymbol:Bitmap;
private var _scanRerollSymbol:Bitmap;

private var _blueprintName:Label;
private var _bpCollectedNumbers:Label;
private var _blueprintCompleteCost:Label;
private var _deepScanCost:Label;
private var _rerollCost:Label;

private var _blueprint:BlueprintVO;
private var _blueprintCost:int;
private var _blueprintProto:IPrototype;

private var _battleRerollVO:BattleRerollVO;

private var _scanCost:Number;
private var _rollCost:Number;

private var _resourceHolder:Sprite;
private var _resAlloySymbol:Bitmap;
private var _resCreditsSymbol:Bitmap;
private var _resEnergySymbol:Bitmap;
private var _resSyntheticsSymbol:Bitmap;
private var _resourceBG:Bitmap;

private var _resAlloyLbl:Label;
private var _resCreditsLbl:Label;
private var _resEnergyLbl:Label;
private var _resSyntheticsLbl:Label;

private var _timeRemainingTitleLbl:Label;
private var _timeRemainingLbl:Label;
private var _timeRemaining:Timer;
private
```

```
var _timeProgressBar:ProgressBar;

private var _colorFilter:ColorMatrixFilter;

private var _scanPrice:int;
private var _rerollPrice:int;

private var _blueprintsCollectedString:String = 'CodeString.Shared.OutOf';
//[[Number.MinValue]]/[[Number.MaxValue]]
private var _timeRemainingText:String = 'CodeString.BattleUserView.TimeRemaining'; //TIME
REMAINING
private var _rerollBtnText:String = 'CodeString.GameOfChance.RerollBtn'; //Reroll
private var _deepScanBtnText:String = 'CodeString.GameOfChance.DeepScanBtn'; //Deep
Scan
private var _noBlueprint:String = 'CodeString.GameOfChance.NoBlueprint'; //BLUEPRINT NOT
FOUND


public function ChanceGameDisplayComponent()
{
super();
onPurchaseComplete = new Signal(String, String);
loadIconSignal = new Signal(String, Function);
rerollClickSignal = new Signal(String, String);
scanClickSignal = new Signal(String);
denyClickSignal = new Signal(String);

_colorFilter = CommonFunctionUtil.getColorMatrixFilter(0x3cf219);

_componentHolder = new Sprite();

var bgRect:Rectangle = new Rectangle(175, 0, 6, 116);
_blueprintBG = UIFactory.getScaleBitmap('LootedBlueprintBMD');
_blueprintBG.scale9Grid = bgRect;
_blueprintBG.width += 100;
_blueprintBG.x = 0; //376;
_blueprintBG.y = 0; //324;

var blueprintBgFrameClass:Class = Class(getDefinitionByName('SelectionFrameBMD'));
_blueprintFrame = new Bitmap(BitmapData(new blueprintBgFrameClass()));
_blueprintFrame.x = 11; //385;
_blueprintFrame.y = 7; //331;

_blueprintShipIcon = new ImageComponent()
_blueprintShipIcon.init(100, 100);

var blueprintNoBPClass:Class = Class(getDefinitionByName('IconNoBlueprintsBMD'));
onBlueprintIconLoaded(BitmapData(new blueprintNoBPClass()));


_blueprintCompleteCost
```

```
= new Label(18, 0xf0f0f0, 114, 25, false);

_blueprintNameBG = PanelFactory.getPanel('BlueprintNameTagBMD');
_blueprintNameBG.x = _blueprintFrame.x + _blueprintFrame.width + 13;
_blueprintNameBG.y = _blueprintFrame.y + _blueprintFrame.height - _blueprintNameBG.height;

_blueprintName = new Label(16, 0x213745, 216, 25);
_blueprintName.x = _blueprintNameBG.x;
_blueprintName.y = _blueprintNameBG.y + 2;
_blueprintName.text = _noBlueprint;

_bpCollectedNumbers = new Label(12, 0xF0F0F0, 10, 25, true, 1);

_denyBtn = ButtonFactory.getCloseButton(437, 0);
_denyBtn.addEventListener(MouseEvent.CLICK, onDenyClick);

_timeRemainingTitleLbl = UIFactory.getLabel(LabelEnum.DEFAULT, 88, 30, _blueprintFrame.x
+ _blueprintFrame.width + 143, _blueprintFrame.y);
_timeRemainingTitleLbl.text = _timeRemainingText;
_timeRemainingLbl = UIFactory.getLabel(LabelEnum.SUBTITLE, 88, 30, _blueprintFrame.x +
_blueprintFrame.width + 143, _timeRemainingTitleLbl.y + _timeRemainingTitleLbl.textHeight +
2);

_timeRemaining = new Timer(1000);
_timeRemaining.addEventListener(TimerEvent.TIMER, onTimeRemainingTimer);

var time:Bitmap = UIFactory.getBitmap(PanelEnum.STATBAR_GREY);
time.width = _timeRemainingTitleLbl.width - 10;
time.height = 15;

_timeProgressBar = new ProgressBar();
_timeProgressBar.init(ProgressBar.HORIZONTAL, time, null, 0.15);
_timeProgressBar.setMinMax(0, 300000);
_timeProgressBar.filters = [_colorFilter];
_timeProgressBar.scaleY = 1.4;
_timeProgressBar.x = _timeRemainingLbl.x + 5;
_timeProgressBar.y = _timeRemainingLbl.y + _timeRemainingLbl.textHeight + 2;

_componentHolder.addChild(_blueprintBG);
_componentHolder.addChild(_blueprintFrame);
_componentHolder.addChild(_blueprintShipIcon);
_componentHolder.addChild(_denyBtn);
_componentHolder.addChild(_timeProgressBar);
_componentHolder.addChild(_timeRemainingTitleLbl);
_componentHolder.addChild(_timeRemainingLbl);
_componentHolder.addChild(_blueprintNameBG);
_componentHolder.addChild(_blueprintName);

addChild(_componentHolder);
}

[PostConstruct]
```

```
public function init( blueprintVO:BlueprintVO ):void
{
if (blueprintVO)
_blueprint = blueprintVO;
}

public function showScanView():void
{
_deepScanBtn = ButtonFactory.getBitmapButton('SquareBuyBtnNeutralBMD', 157, 5,
_deepScanBtnText, 0xf7c78b, 'SquareBuyBtnRollOverBMD', 'SquareBuyBtnSelectedBMD');
_deepScanBtn.x = _blueprintFrame.x + _blueprintFrame.width + 13;
_deepScanBtn.y = _blueprintFrame.y + 2
_deepScanBtn.label.fontSize = 22;
_deepScanBtn.addEventListener(MouseEvent.CLICK, onDeepScanClick);
_componentHolder.addChild(_deepScanBtn);

_deepScanCost = UIFactory.getLabel(LabelEnum.DEFAULT, 114, 25, _deepScanBtn.x + 4,
_deepScanBtn.y + 26);
_deepScanCost.fontSize = 18;
_deepScanCost.text = String(_scanCost);
_componentHolder.addChild(_deepScanCost);

_scanRerollSymbol = UIFactory.getBitmap('KalganSymbolBMD');
_scanRerollSymbol.x = _deepScanBtn.x + 7;
_scanRerollSymbol.y = _deepScanBtn.y + 24;
_componentHolder.addChild(_scanRerollSymbol);

if (_battleRerollVO.timeRemaining > 0 && !_timeRemaining.running)
_timeRemaining.start();

}

public function showGainedResourcesView( rerollVO:BattleRerollVO ):void
{
_blueprintBG.visible = false;
_blueprintFrame.visible = false;
_denyBtn.visible = false;
_blueprintShipIcon.visible = false;

if (_rerollBtn)
{
_rerollBtn.visible = false;
_rerollCost.visible = false;
_scanRerollSymbol.visible = false;
}

if (_deepScanBtn)
{
_deepScanBtn.visible = false;
_deepScanCost.visible
```

```
= false;
_scanRerollSymbol.visible = false;
}

if (_timeRemainingLbl)
{
_timeRemainingLbl.visible = false;
_timeRemainingTitleLbl.visible = false;
_timeProgressBar.visible = false;
}

getResourceIcons(rerollVO.creditsReward, rerollVO.alloyReward, rerollVO.energyReward,
rerollVO.syntheticReward);
}

private function getResourceIcons( credGained:int, alloyGained:int, energyGained:int,
synthGained:int ):void
{
_resourceBG = UIFactory.getBitmap('LootedResourceContainerBMD');
_resourceBG.x = 103;
_resourceBG.y = 0;

_resourceHolder = new Sprite();
_resourceHolder.x = 109;
_resourceHolder.y = 11;

_resCreditsSymbol = PanelFactory.getPanel('LootedResourceCreditsBMD');

_resEnergySymbol = PanelFactory.getPanel('LootedResourceEnergyBMD');
_resEnergySymbol.x = _resCreditsSymbol.x + 179;
_resEnergySymbol.y = _resCreditsSymbol.y;

_resAlloySymbol = PanelFactory.getPanel('LootedResourceAlloyBMD');
_resAlloySymbol.x = _resCreditsSymbol.x;
_resAlloySymbol.y = _resCreditsSymbol.y + 50;

_resSyntheticsSymbol = PanelFactory.getPanel('LootedResourceSyntheticsBMD');
_resSyntheticsSymbol.x = _resEnergySymbol.x;
_resSyntheticsSymbol.y = _resAlloySymbol.y;


_resCreditsLbl = new Label(13, 0xffdd3d, 140, 30, true, 1);
_resCreditsLbl.x = _resCreditsSymbol.width * 0.5 + _resCreditsSymbol.x - 35;
_resCreditsLbl.y = _resCreditsSymbol.height * 0.5 + _resCreditsSymbol.y - 4;
_resCreditsLbl.constrictTextToSize = false;
_resCreditsLbl.align = TextFormatAlign.LEFT;
_resCreditsLbl.text = StringUtil.commaFormatNumber(credGained);
_resCreditsLbl.letterSpacing = 1.5;


_resAlloyLbl
```

```
= new Label(13, 0xffdd3d, 140, 30, true, 1);
_resAlloyLbl.x = _resAlloySymbol.width * 0.5 + _resAlloySymbol.x - 35;
_resAlloyLbl.y = _resAlloySymbol.height * 0.5 + _resAlloySymbol.y - 4;
_resAlloyLbl.constrictTextToSize = false;
_resAlloyLbl.align = TextFormatAlign.LEFT;
_resAlloyLbl.text = StringUtil.commaFormatNumber(alloyGained);
_resAlloyLbl.letterSpacing = 1.5;


_resSyntheticsLbl = new Label(13, 0xffdd3d, 140, 30, true, 1);
_resSyntheticsLbl.x = _resSyntheticsSymbol.width * 0.5 + _resSyntheticsSymbol.x - 35;
_resSyntheticsLbl.y = _resSyntheticsSymbol.height * 0.5 + _resSyntheticsSymbol.y - 4;
_resSyntheticsLbl.constrictTextToSize = false;
_resSyntheticsLbl.align = TextFormatAlign.LEFT;
_resSyntheticsLbl.text = StringUtil.commaFormatNumber(synthGained);
_resSyntheticsLbl.letterSpacing = 1.5;


_resEnergyLbl = new Label(13, 0xffdd3d, 140, 30, true, 1);
_resEnergyLbl.x = _resEnergySymbol.width * 0.5 + _resEnergySymbol.x - 35;
_resEnergyLbl.y = _resEnergySymbol.height * 0.5 + _resEnergySymbol.y - 4;
_resEnergyLbl.constrictTextToSize = false;
_resEnergyLbl.align = TextFormatAlign.LEFT;
_resEnergyLbl.text = StringUtil.commaFormatNumber(energyGained);
_resEnergyLbl.letterSpacing = 1.5;

_resourceHolder.addChild(_resCreditsSymbol);
_resourceHolder.addChild(_resAlloySymbol);
_resourceHolder.addChild(_resSyntheticsSymbol);
_resourceHolder.addChild(_resEnergySymbol);
_resourceHolder.addChild(_resCreditsLbl);
_resourceHolder.addChild(_resAlloyLbl);
_resourceHolder.addChild(_resSyntheticsLbl);
_resourceHolder.addChild(_resEnergyLbl);


_componentHolder.addChild(_resourceBG);
_componentHolder.addChild(_resourceHolder);
}

public function showBlueprint( blueprintProtoName:String, blueprintVO:BlueprintVO,
bpAsset:AssetVO, hardCurrencyCost:int ):void
{
_blueprintGlow = PanelFactory.getPanel('BlueprintGlowBMD');
_blueprintGlow.x = -7;
_blueprintGlow.y = -10;
_blueprintGlow.alpha = 0;

_purchaseBlueprintBtn = ButtonFactory.getBitmapButton('SquareBuyBtnNeutralBMD', 166, 23,
'CodeString.Dialogue.Complete', 0xf7c78b, 'SquareBuyBtnRollOverBMD',
'SquareBuyBtnSelectedBMD');
```

```
_purchaseBlueprintBtn.x = _blueprintFrame.x + _blueprintFrame.width + 2;
_purchaseBlueprintBtn.y = _blueprintFrame.y + 12;
_purchaseBlueprintBtn.label.fontSize = 22;
_purchaseBlueprintBtn.addEventListener(MouseEvent.CLICK, onBlueprintPurchase, false, 0,
true);

if (_battleRerollVO && _battleRerollVO.isReroll && !_battleRerollVO.hasPaid)
{
_rerollBtn = ButtonFactory.getBitmapButton('SquareBuyBtnNeutralBMD',
_purchaseBlueprintBtn.x + 130, _purchaseBlueprintBtn.y, _rerollBtnText, 0xf7c78b,
'SquareBuyBtnRollOverBMD', 'SquareBuyBtnSelectedBMD');
_rerollBtn.label.fontSize = 22;
_rerollBtn.addEventListener(MouseEvent.CLICK, onRerollClick, false, 0, true);

_rerollCost = UIFactory.getLabel(LabelEnum.DEFAULT, 114, 25, _rerollBtn.x + 4, _rerollBtn.y +
26);
_rerollCost.fontSize = 18;
_rerollCost.text = String(_rollCost);

_scanRerollSymbol = UIFactory.getBitmap('KalganSymbolBMD');
_scanRerollSymbol.x = _rerollBtn.x + 7;
_scanRerollSymbol.y = _rerollBtn.y + 24;

_timeRemainingTitleLbl.x = _rerollBtn.x + _rerollBtn.width + 2;
_timeRemainingTitleLbl.y = _rerollBtn.y - 1;

_timeRemainingLbl.x = _timeRemainingTitleLbl.x;
_timeRemainingLbl.y = _timeRemainingTitleLbl.y + _timeRemainingTitleLbl.textHeight + 2;

_timeProgressBar.x = _timeRemainingLbl.x + 5;
_timeProgressBar.y = _timeRemainingLbl.y + _timeRemainingLbl.textHeight + 2;

if (_battleRerollVO.timeRemaining > 0 && !_timeRemaining.running)
_timeRemaining.start();
}

_blueprintCompleteCost.align = TextFormatAlign.CENTER;
_blueprintCompleteCost.constrictTextToSize = false;
_blueprintCompleteCost.x = _purchaseBlueprintBtn.x + 4;
_blueprintCompleteCost.y = _purchaseBlueprintBtn.y + 26;

_blueprintPremiumSymbol = PanelFactory.getPanel('KalganSymbolBMD');
_blueprintPremiumSymbol.x = _purchaseBlueprintBtn.x + 7;
_blueprintPremiumSymbol.y = _purchaseBlueprintBtn.y + 24;

_blueprintProto = getBlueprintPrototype(blueprintProtoName);
if (!_blueprintProto)
_blueprintProto = getBlueprintPrototype(blueprintVO.name);

if
```

```
(bpAsset.largeImage == '')
loadIconSignal.dispatch(bpAsset.mediumImage, onBlueprintIconLoaded);
else
loadIconSignal.dispatch(bpAsset.largeImage, onBlueprintIconLoaded);

var rarity:String;
if (_blueprintProto)
rarity = _blueprintProto.getValue('rarity');
else if (blueprintVO)
rarity = blueprintVO.prototype.getValue('rarity');

var bpLabelColor:uint = CommonFunctionUtil.getRarityColor(rarity);
_blueprintFrame.filters = [CommonFunctionUtil.getRarityGlow(rarity)];

_blueprintName.textColor = bpLabelColor;
_blueprintName.constrictTextToSize = false;
_blueprintName.align = TextFormatAlign.CENTER;
_blueprintName.text = bpAsset.visibleName;
_blueprintName.letterSpacing = 1.5;

var numCollected:int = 0;
if (_blueprint)
numCollected = _blueprint.partsCollected;
else
++numCollected;

_bpCollectedNumbers.constrictTextToSize = false;
_bpCollectedNumbers.autoSize = TextFieldAutoSize.LEFT;
if (_blueprintProto)
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':numCollected, '[[Number.MaxValue]]':_blueprintProto.getValue('parts')});
else if (blueprintVO)
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':numCollected,
'[[Number.MaxValue]]':blueprintVO.prototype.getValue('parts')});
_bpCollectedNumbers.letterSpacing = 1;

if (_blueprint)
{
if (_blueprint.complete)
{
_purchaseBlueprintBtn.visible = false;
_blueprintCompleteCost.visible = false;
_blueprintPremiumSymbol.visible = false;
onDenyClick(null);
_blueprintNameBG.x = 139;
_blueprintNameBG.y = _blueprintBG.y + 43;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x
```

```
= _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;

onBlueprintGlowFadeOut();
} else
{
_purchaseBlueprintBtn.visible = true;
_blueprintCompleteCost.visible = true;
_blueprintPremiumSymbol.visible = true;
_blueprintCost = hardCurrencyCost;
_blueprintCompleteCost.text = String(_blueprintCost);

_blueprintNameBG.x = 139;
_blueprintNameBG.y = 79;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;
onBlueprintGlowFadeOut();
}
} else
{
_blueprintNameBG.x = 139;
_blueprintNameBG.y = 79;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;
}

if (_battleRerollVO)
{
_componentHolder.addChild(_blueprintGlow);
_componentHolder.addChild(_bpCollectedNumbers);

_componentHolder.addChild(_purchaseBlueprintBtn);
if (_rerollBtn)
_componentHolder.addChild(_rerollBtn);
if (_rerollCost)
_componentHolder.addChild(_rerollCost);
if (_scanRerollSymbol)
_componentHolder.addChild(_scanRerollSymbol);
_componentHolder.addChild(_blueprintCompleteCost);
_componentHolder.addChild(_blueprintPremiumSymbol);
}

if
```

```
(_blueprint)
{
if (_blueprint.complete && _componentHolder.contains(_purchaseBlueprintBtn))
_componentHolder.removeChild(_purchaseBlueprintBtn);
}
}

private function onBlueprintIconLoaded( asset:BitmapData ):void
{
if (asset && _blueprintShipIcon)
{
_blueprintShipIcon.clearBitmap();
_blueprintShipIcon.onImageLoaded(asset);
_blueprintShipIcon.x = (_blueprintFrame.x + _blueprintFrame.width * 0.5) -
(_blueprintShipIcon.width * 0.5);
_blueprintShipIcon.y = (_blueprintFrame.y + _blueprintFrame.height * 0.5) -
(_blueprintShipIcon.height * 0.5);
}
}

private function onBlueprintGlowFadeOut():void
{
TweenLite.to(_blueprintGlow, 1.0, {alpha:1.0, ease:Quad.easeOut,
onComplete:onBlueprintGlowFadeIn, overwrite:0});
}

private function onBlueprintGlowFadeIn():void
{
TweenLite.to(_blueprintGlow, 1.0, {alpha:0.0, ease:Quad.easeIn,
onComplete:onBlueprintGlowFadeOut, overwrite:0});
}

private function onBlueprintPurchase( e:MouseEvent ):void
{
if (_blueprint && !_blueprint.complete)
{
if (CurrentUser.wallet.premium >= _blueprintCost)
{
onPurchaseComplete.dispatch(_blueprint.id, _battleRerollVO.battleKey);
if (_rerollBtn)
_scanRerollSymbol.visible = _rerollCost.visible = _rerollBtn.visible = false;
_timeProgressBar.visible = _timeRemainingLbl.visible = false;
_denyBtn.visible = false;
_componentHolder.visible = false;
_purchaseBlueprintBtn.visible = false;
_blueprintCompleteCost.visible = false;
_blueprintPremiumSymbol.visible = false;
if (_bpCollectedNumbers)
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':_blueprint.totalParts, '[[Number.MaxValue]]':_blueprint.totalParts});
onBlueprintGlowFadeOut();
```

```
} else
CommonFunctionUtil.popPaywall();
}
}

public function tooltip( prototype:IPrototype ):String
{
return getSelectionInfo(prototype, StringUtil.getTooltip);
}

public function getTooltip():String
{
var proto:IPrototype = getResearchPrototypeByName(_blueprintProto.getValue('key'));
return String(getSelectionInfo(proto, StringUtil.getTooltip));
}

public function hideBlueprint():void
{
if (_blueprintShipIcon)
_blueprintShipIcon.clearBitmap();

if (_blueprintName)
_blueprintName.text = '';

if (_purchaseBlueprintBtn)
{
_purchaseBlueprintBtn.visible = false;
_blueprintPremiumSymbol.visible = false;
}
}

public function updateBlueprint( blueprintProtoName:String, blueprintVO:BlueprintVO,
bpAsset:AssetVO, hardCurrencyCost:int ):void
{
_denyBtn.visible = false;

if (_rerollBtn)
{
_rerollBtn.visible = false;
_rerollCost.visible = false;
_scanRerollSymbol.visible = false;
}

if (_deepScanBtn)
{
_deepScanBtn.visible = false;
_deepScanCost.visible = false;
_scanRerollSymbol.visible = false;
}
```

```
if (_timeRemainingLbl)
{
_timeRemainingLbl.visible = false;
_timeProgressBar.visible = false;
}

hideBlueprint();

if (!_battleRerollVO.isReroll)
showBlueprint(blueprintProtoName, blueprintVO, bpAsset, hardCurrencyCost);


_blueprintProto = getBlueprintPrototype(blueprintProtoName);
if (!_blueprintProto)
_blueprintProto = getBlueprintPrototype(blueprintVO.name);

if (bpAsset.largeImage == '')
loadIconSignal.dispatch(bpAsset.mediumImage, onBlueprintIconLoaded);
else
loadIconSignal.dispatch(bpAsset.largeImage, onBlueprintIconLoaded);

var rarity:String;
if (_blueprintProto)
rarity = _blueprintProto.getValue('rarity');
else if (blueprintVO)
rarity = blueprintVO.prototype.getValue('rarity');

var bpLabelColor:uint = CommonFunctionUtil.getRarityColor(rarity);
_blueprintFrame.filters = [CommonFunctionUtil.getRarityGlow(rarity)];

_blueprintName.text = bpAsset.visibleName;

var numCollected:int = 0;
if (_blueprint)
numCollected = _blueprint.partsCollected;
else
++numCollected;

if (_blueprintProto)
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':numCollected, '[[Number.MaxValue]]':_blueprintProto.getValue('parts')});
else if (blueprintVO)
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':numCollected,
'[[Number.MaxValue]]':blueprintVO.prototype.getValue('parts')});

if (_blueprint)
{
if
```

```
(_blueprint.complete)
{
_purchaseBlueprintBtn.visible = false;
_blueprintCompleteCost.visible = false;
_blueprintPremiumSymbol.visible = false;
_blueprintNameBG.x = 139;
_blueprintNameBG.y = _blueprintBG.y + 43;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;

onBlueprintGlowFadeOut();
} else
{
_purchaseBlueprintBtn.visible = true;
_blueprintCompleteCost.visible = true;
_blueprintPremiumSymbol.visible = true;
_blueprintCost = hardCurrencyCost;
_blueprintCompleteCost.text = String(_blueprintCost);

_blueprintNameBG.x = 139;
_blueprintNameBG.y = 79;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;
onBlueprintGlowFadeOut();
}
} else
{
_blueprintNameBG.x = 139;
_blueprintNameBG.y = 79;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;
}


}

private function onDenyClick( e:MouseEvent ):void
{
_denyBtn.visible
```

```
= false;

if (_rerollBtn)
{
_rerollBtn.visible = false;
_rerollCost.visible = false;
_scanRerollSymbol.visible = false;
}

if (_deepScanBtn)
{
_deepScanBtn.visible = false;
_deepScanCost.visible = false;
_scanRerollSymbol.visible = false;
}

if (_timeProgressBar)
{
_timeRemainingTitleLbl.visible = _timeRemainingLbl.visible = _timeProgressBar.visible = false;
}

if (e)
{
_componentHolder.visible = false;
if(_battleRerollVO != null)
denyClickSignal.dispatch(_battleRerollVO.battleKey);
}
}

private function onTimeRemainingTimer( e:TimerEvent ):void
{
if (_battleRerollVO.timeRemaining <= 0)
{
_timeRemainingLbl.visible = false;
_timeRemaining.stop();
onDenyClick(null);
} else
{
_timeRemainingLbl.text = StringUtil.getBuildTime(_battleRerollVO.timeRemaining / 1000);
if (_timeProgressBar)
_timeProgressBar.amount = (_battleRerollVO.timeRemaining);
}
}

private function onRerollClick( e:MouseEvent ):void
{
if (CurrentUser.wallet.premium >= _rollCost)
{
_timeRemaining.stop();
onDenyClick(null);
rerollClickSignal.dispatch(_battleRerollVO.battleKey,
```

```
_blueprint.prototype.name);

} else
CommonFunctionUtil.popPaywall();

}

private function onDeepScanClick( e:MouseEvent ):void
{
if (CurrentUser.wallet.premium >= _scanCost)
{
_timeRemaining.stop();
onDenyClick(null);
scanClickSignal.dispatch(_battleRerollVO.battleKey);
} else
CommonFunctionUtil.popPaywall();
}

override public function get width():Number { return _componentHolder.width; }
override public function get height():Number { return _componentHolder.height; }

public function setTimeRemaining( time:Number ):void { _timeRemainingLbl.text = String(time); }
public function getTimeRemainingText():Number { return Number(_timeRemainingLbl.text); }

public function get enabled():Boolean { return false; }
public function set enabled( value:Boolean ):void {}

public function get blueprintShipIcon():ImageComponent { return _blueprintShipIcon; }
public function set blueprintShipIcon( value:ImageComponent ):void { _blueprintShipIcon =
value; }

public function get battleRerollVO():BattleRerollVO { return _battleRerollVO; }
public function set battleRerollVO( value:BattleRerollVO ):void { _battleRerollVO = value; }

public function get timeRemainingLbl():Label { return _timeRemainingLbl; }
public function set timeRemainingLbl( value:Label ):void { _timeRemainingLbl = value; }

public function get cargoFull():Boolean { return _cargoFull; }
public function set cargoFull( value:Boolean ):void { _cargoFull = value; }

public function get componentHolder():Sprite { return _componentHolder; }
public function set componentHolder( value:Sprite ):void { _componentHolder = value; }

public function get percent():Number { return _timeProgressBar.amount; }
public function set percent( v:Number ):void { _timeProgressBar.amount = v; }

public function set scanCost( value:Number ):void { _scanCost = value; }
public function set rollCost( value:Number ):void { _rollCost = value; }

public function destroy():void
{
```

```
_blueprintFrame = null;
_blueprintBG = null;
_blueprintPremiumSymbol = null;
_blueprintNameBG = null;
_scanRerollSymbol = null;
_resAlloySymbol = null;
_resCreditsSymbol = null;
_resEnergySymbol = null;
_resSyntheticsSymbol = null;
_resourceBG = null;
_blueprint = null;

_blueprintProto = null;
_battleRerollVO = null;
_resourceHolder = null;
_componentHolder = null;

onPurchaseComplete.removeAll();
loadIconSignal.removeAll();
rerollClickSignal.removeAll();
scanClickSignal.removeAll();
denyClickSignal.removeAll();

onPurchaseComplete = null;
loadIconSignal = null;
rerollClickSignal = null;
scanClickSignal = null;
denyClickSignal = null;

if (_blueprintGlow)
TweenLite.killTweensOf(_blueprintGlow);

_blueprintGlow = null;

if (_blueprintShipIcon)
ObjectPool.give(_blueprintShipIcon);

_blueprintShipIcon = null;

if (_purchaseBlueprintBtn)
_purchaseBlueprintBtn.destroy();

_purchaseBlueprintBtn = null;

if (_blueprintName)
_blueprintName.destroy();

_blueprintName = null;

if
```

```
(_bpCollectedNumbers)
_bpCollectedNumbers.destroy();

_bpCollectedNumbers = null;

if (_blueprintCompleteCost)
_blueprintCompleteCost.destroy();

_blueprintCompleteCost = null;

if (_purchaseBlueprintBtn)
{
_purchaseBlueprintBtn.destroy();
_purchaseBlueprintBtn = null;
}

if (_rerollBtn)
{
_rerollBtn.destroy();
_rerollBtn = null;
}

if (_rerollCost)
{
_rerollCost.destroy();
_rerollCost = null;
}

if (_deepScanBtn)
{
_deepScanBtn.destroy();
_deepScanBtn = null;
}

if (_deepScanCost)
{
_deepScanCost.destroy();
_deepScanCost = null;
}

if (_denyBtn)
{
_denyBtn.destroy();
_denyBtn = null;
}


if (_resAlloyLbl)
{
_resAlloyLbl.destroy();
_resAlloyLbl
```

```
= null;
}

if (_resCreditsLbl)
{
_resCreditsLbl.destroy();
_resCreditsLbl = null;
}

if (_resEnergyLbl)
{
_resEnergyLbl.destroy();
_resEnergyLbl = null;
}

if (_resSyntheticsLbl)
{
_resSyntheticsLbl.destroy();
_resSyntheticsLbl = null;
}

if (_timeRemainingLbl)
{
_timeRemainingLbl.destroy();
_timeRemainingLbl = null;
}

_timeRemaining.stop();

if (_timeProgressBar)
_timeProgressBar.destroy();

_timeProgressBar = null;
}

}
}

-----------------------------------
File 791: igw\com\ui\modal\battle\chance\GameOfChanceListView.as
package com.ui.modal.battle.chance
{
import com.model.asset.AssetVO;
import com.model.battle.BattleRerollVO;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.IGameOfChancePresenter;
import com.presenter.shared.IUIPresenter;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import
```

```
com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.battlelog.BattleLogEntry;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.shared.ObjectPool;

public class GameOfChanceListView extends View
{
private var _bg:DefaultWindowBG;
private var _chances:Vector.<ChanceGameDisplayComponent>;
private var _holder:Sprite;
private var _maxHeight:int;
private var _scrollbar:VScrollbar;
private var _scrollRect:Rectangle;
private var _tooltips:Tooltips;

private var _titleText:String = 'CodeString.PendingScans.Title'; //PENDING SCANS

[PostConstruct]
override public function init():void
{
super.init();

_chances = new Vector.<ChanceGameDisplayComponent>;

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(500, 425);
_bg.addTitle(_titleText, 135);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_holder = new Sprite();
_holder.x = 26;
_holder.y = 50;
_maxHeight = 0;

_scrollRect = new Rectangle(0, 0, 580, 395);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect;

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = 492;
var scrollbarYPos:Number = 57;
_scrollbar.init(7, _scrollRect.height - 10, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
```

```
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 28.25;

addChild(_bg);
addChild(_scrollbar);
addChild(_holder);

presenter.addAvailableRerollUpdatedListener(onAvailableRerollsUpdated);
presenter.addRerollFromRerollCallback(onRerollUpdated);
presenter.addRerollFromScanCallback(onRerollUpdated);

var rerolls:Vector.<BattleRerollVO> = presenter.getAvailableRerolls();

//var vo:BattleRerollVO = new BattleRerollVO("sector.2.battle.14.1368552417881",
"player.502.blueprint.1", 300000);
for each (var vo:BattleRerollVO in rerolls)
onAvailableRerollsUpdated(vo);

addEffects();
effectsIN();
}

public function onRerollUpdated( vo:BattleRerollVO ):void
{
for each (var chanceComponent:ChanceGameDisplayComponent in _chances)
{
if (chanceComponent.battleRerollVO == vo)
{
if (vo.blueprintPrototype != '')
updateBlueprint(chanceComponent, vo);
else
{
chanceComponent.x -= 48;
chanceComponent.showGainedResourcesView(vo);
}
}
}
}

private function onAvailableRerollsUpdated( vo:BattleRerollVO ):void
{
if (!vo.isReroll && !vo.hasPaid)
createScanView(vo);
else if (vo.isReroll && !vo.hasPaid)
createRerollView(vo);

layout();
}

private
```

```
function createScanView( vo:BattleRerollVO ):void
{
var chanceComp:ChanceGameDisplayComponent = new ChanceGameDisplayComponent();
chanceComp.battleRerollVO = vo;
chanceComp.scanCost = presenter.getConstantPrototypeValueByName('rerollLootPrice');
chanceComp.getBlueprintPrototype = presenter.getBlueprintPrototypeByName;
chanceComp.getResearchPrototypeByName = presenter.getResearchPrototypeByName;
chanceComp.getSelectionInfo = getSelectionInfo
chanceComp.scanClickSignal.add(onScanClick);
chanceComp.denyClickSignal.add(onDenyClick);
chanceComp.showScanView();
_chances.push(chanceComp);
_holder.addChild(chanceComp);
}

private function createRerollView( vo:BattleRerollVO ):void
{
if (vo && _holder)
{
var blueprintVO:BlueprintVO;
if (vo.recievedBlueprintPrototype)
blueprintVO = presenter.getBlueprintByID(vo.recievedBlueprintPrototype);
else if (vo.blueprintPrototype)
blueprintVO = presenter.getBlueprintByName(vo.blueprintPrototype);
var bpAsset:AssetVO = presenter.getAssetVO(blueprintVO.uiAsset);

if (blueprintVO)
{
var chanceComp:ChanceGameDisplayComponent = new ChanceGameDisplayComponent();
chanceComp.battleRerollVO = vo;
chanceComp.loadIconSignal.add(presenter.loadIcon);
chanceComp.onPurchaseComplete.add(onBlueprintPurchase);
chanceComp.denyClickSignal.add(onDenyClick);
chanceComp.rollCost = presenter.getConstantPrototypeValueByName('rerollItemPrice');

chanceComp.getBlueprintPrototype = presenter.getBlueprintPrototypeByName;
chanceComp.getResearchPrototypeByName = presenter.getResearchPrototypeByName;
chanceComp.getSelectionInfo = getSelectionInfo

_tooltips.addTooltip(chanceComp.blueprintShipIcon, this, chanceComp.getTooltip);
chanceComp.init(blueprintVO);
chanceComp.rerollClickSignal.add(onRerollClick);
chanceComp.showBlueprint(vo.blueprintPrototype, blueprintVO, bpAsset,
presenter.getBlueprintHardCurrencyCost(blueprintVO, blueprintVO.partsRemaining));
_chances.push(chanceComp);
_holder.addChild(chanceComp);
}
}
}

private
```

```
function updateBlueprint( chanceComponent:ChanceGameDisplayComponent,
vo:BattleRerollVO ):void
{
var blueprintVO:BlueprintVO;
if (vo.blueprintPrototype)
blueprintVO = presenter.getBlueprintByName(vo.blueprintPrototype);
if (vo.recievedBlueprintPrototype && !blueprintVO)
blueprintVO = presenter.getBlueprintByID(vo.recievedBlueprintPrototype);

var bpAsset:AssetVO = presenter.getAssetVO(blueprintVO.uiAsset);

chanceComponent.battleRerollVO = vo;
chanceComponent.loadIconSignal.add(presenter.loadIcon);
chanceComponent.onPurchaseComplete.add(onBlueprintPurchase);
chanceComponent.rollCost = presenter.getConstantPrototypeValueByName('rerollItemPrice');
//Get BP tooltip
_tooltips.addTooltip(chanceComponent.blueprintShipIcon, chanceComponent,
chanceComponent.getTooltip);
chanceComponent.init(blueprintVO);
chanceComponent.rerollClickSignal.add(onRerollClick);
chanceComponent.updateBlueprint(vo.blueprintPrototype, blueprintVO, bpAsset,
presenter.getBlueprintHardCurrencyCost(blueprintVO, blueprintVO.partsRemaining));
}

protected function layout():void
{
var len:uint = _chances.length;
var selection:ChanceGameDisplayComponent;
var yPos:int = 00;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _chances[i];
selection.y = yPos;
_maxHeight += selection.height;
yPos += selection.height;
}
_scrollbar.updateScrollableHeight(_maxHeight);
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

private function orderItems( itemOne:BattleLogEntry, itemTwo:BattleLogEntry ):Number
{

if (!itemOne)
return
```

```
-1;
if (!itemTwo)
return 1;

var timeOccurredOne:Number = itemOne.timeOccurred;
var timeOccurredTwo:Number = itemTwo.timeOccurred;

if (timeOccurredOne < timeOccurredTwo)
return -1;
else if (timeOccurredOne > timeOccurredTwo)
return 1;

return 0;
}

private function getSelectionInfo( prototype:IPrototype, info:Function ):*
{
var selectionInfo:*;
if (prototype)
{
var reqBuildClass:String = prototype.getUnsafeValue('requiredBuildingClass');
if (reqBuildClass)
{
var proto:IPrototype;
if (reqBuildClass == 'ShipDesignFacility')
{
proto = presenter.getShipPrototype(prototype.getValue('referenceName'));
if (proto)
{
selectionInfo = info(proto.getValue('type'), proto);
} else
{
proto = presenter.getShipPrototype(prototype.getValue('referenceName'));
if (proto)
selectionInfo = info(proto.getValue('type'), proto);
}

} else if (reqBuildClass == 'CommandCenter')
{
selectionInfo = info(prototype.getValue('type'), prototype);
} else if (reqBuildClass == 'WeaponsDesignFacility')
{
proto = presenter.getModulePrototypeByName(prototype.getValue('referenceName'));
selectionInfo = info(proto.getValue('type'), proto);
} else
{
proto = presenter.getModulePrototypeByName(prototype.getValue('referenceName'));
selectionInfo = info(proto.getValue('type'), proto);
}
} else
selectionInfo
```

```
= info(prototype.getValue('type'), prototype);
}

return selectionInfo;
}

private function onRerollClick( battleID:String, name:String ):void
{
presenter.removeBlueprintByName(name);
presenter.purchaseReroll(battleID);
}

private function onScanClick( battleID:String ):void
{
presenter.purchaseDeepScan(battleID);
}

private function onBlueprintPurchase( blueprintID:String, battleID:String ):void
{
var vo:BlueprintVO = presenter.getBlueprintByID(blueprintID);
presenter.purchaseBlueprint(vo, vo.partsRemaining);
presenter.removeRerollFromAvailable(battleID);
}

private function onDenyClick( battleID:String ):void
{
presenter.removeRerollFromAvailable(battleID);
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IGameOfChancePresenter ):void { _presenter = value; }
public function get presenter():IGameOfChancePresenter { return
IGameOfChancePresenter(_presenter); }
[Inject]
public function set tooltips( value:Tooltips ):void { _tooltips = value; }

override public function destroy():void
{
presenter.removeAvailableRerollUpdatedListener(onAvailableRerollsUpdated);
presenter.removeRerollFromRerollCallback(onRerollUpdated);
presenter.removeRerollFromScanCallback(onRerollUpdated);

super.destroy();

ObjectPool.give(_bg);
_bg = null;

_holder
```

```
= null;

_scrollbar.destroy();
_scrollbar = null;

_maxHeight = 0;

_tooltips.removeTooltip(null, this);
_tooltips = null;

_chances = null;


}
}
}

-----------------------------------
File 792: igw\com\ui\modal\battlelog\BattleLogDetailView.as
package com.ui.modal.battlelog
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.battlelog.BattleLogPlayerInfoVO;
import com.model.battlelog.BattleLogVO;
import com.model.player.CurrentUser;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.filters.GlowFilter;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;
import org.shared.ObjectPool;

public
```

```
class BattleLogDetailView extends View
{
private var _bg:DefaultWindowBG;

private var _winners:Vector.<BattleLogPlayerPanel>;
private var _losers:Vector.<BattleLogPlayerPanel>;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _holder:Sprite;

private var _battleLog:BattleLogVO;

private var _alloyCount:Label;
private var _creditsCount:Label;
private var _energyCount:Label;
private var _syntheticsCount:Label;

private var _blueprintTitle:Label;
private var _blueprintName:Label;
private var _quoteText:Label;

private var _victorsHeading:Label;
private var _losersHeading:Label;

private var _blueprintFrame:Bitmap;
private var _blueprintWeb:Bitmap;
private var _blueprintBg:Bitmap;
private var _alloyFrame:Bitmap;
private var _creditsFrame:Bitmap;
private var _energyFrame:Bitmap;
private var _syntheticsFrame:Bitmap;

private var _blueprintImage:ImageComponent;

private var _victorBG:ScaleBitmap;
private var _defeatedBG:ScaleBitmap;
private var _quoteFrame:ScaleBitmap;

private var _currentUserHasWon:Boolean;

private var _currentUsersOutcome:int;

private var _blueprint:IPrototype;

private static var WINNER:int = 0;
private static var LOSER:int = 1;
private
```

```
static var DRAW:int = 2;

private var _titleText:String = 'CodeString.BattleLogs.LogTitle'; //Battlelog
private var _victorText:String = 'CodeString.BattleLog.Victor'; //Victor
private var _defeatedText:String = 'CodeString.BattleLog.Defeated'; //Defeated
private var _drawText:String = 'CodeString.BattleLog.DrawTitle'; //Draw
private var _blueprintTitleText:String = 'CodeString.BattleLog.BlueprintTitle'; //BLUEPRINT

[PostConstruct]
override public function init():void
{
super.init();

_winners = new Vector.<BattleLogPlayerPanel>;
_losers = new Vector.<BattleLogPlayerPanel>;

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(582, 450);
_bg.addTitle(_titleText, 139);
_bg.x -= 21;
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);
addChild(_bg);

_holder = new Sprite();
_holder.x = 10;
_holder.y = 60;
_maxHeight = 0;

_scrollRect = new Rectangle(0, 0, 550, 413);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect;

_blueprintImage = ObjectPool.get(ImageComponent);
_blueprintImage.init(2000, 2000);

_victorBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_victorBG.width = 539;
_victorBG.height = 33;

_defeatedBG = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_defeatedBG.width = 539;
_defeatedBG.height = 33;

_quoteFrame = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_quoteFrame.width = 532;
_quoteFrame.height = 33;

_blueprintBg = UIFactory.getBitmap('BattleLogBlueprintBGBMD');

_blueprintFrame = UIFactory.getBitmap('BattleLogBlueprintFrameBMD');

_blueprintWeb
```

```
= UIFactory.getBitmap('BattleLogBlueprintWebBMD');

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = 550;
var scrollbarYPos:Number = 78;
_scrollbar.init(7, _scrollRect.height - 32, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);

_alloyFrame = UIFactory.getBitmap('LootedResourceAlloyBMD');
_creditsFrame = UIFactory.getBitmap('LootedResourceCreditsBMD');
_energyFrame = UIFactory.getBitmap('LootedResourceEnergyBMD');
_syntheticsFrame = UIFactory.getBitmap('LootedResourceSyntheticsBMD');

_creditsCount = new Label(16, 0xf04c4c, 108, 30, true, 1);
_creditsCount.constrictTextToSize = false;
_creditsCount.align = TextFormatAlign.CENTER;
_creditsCount.letterSpacing = 1.5;

_alloyCount = new Label(16, 0xf04c4c, 108, 108, true, 1);
_alloyCount.constrictTextToSize = false;
_alloyCount.align = TextFormatAlign.CENTER;
_alloyCount.letterSpacing = 1.5;

_energyCount = new Label(16, 0xf04c4c, 108, 108, true, 1);
_energyCount.constrictTextToSize = false;
_energyCount.align = TextFormatAlign.CENTER;

_syntheticsCount = new Label(16, 0xf04c4c, 108, 108, true, 1);
_syntheticsCount.constrictTextToSize = false;
_syntheticsCount.align = TextFormatAlign.CENTER;
_syntheticsCount.letterSpacing = 1.5;

_quoteText = new Label(21, 0x7ca5fd, 512, 33, true);
_quoteText.align = TextFormatAlign.CENTER;

_victorsHeading = new Label(22, 0xffdd3d, 515, 33);
_victorsHeading.align = TextFormatAlign.LEFT;
_victorsHeading.text = _victorText;

_losersHeading = new Label(22, 0xfe4f4f, 515, 33);
_losersHeading.align = TextFormatAlign.RIGHT;

_blueprintName = new Label(14, 0xf0f0f0, 189, 25, true, 1);
_blueprintName.align = TextFormatAlign.CENTER;

_blueprintTitle = new Label(20, 0xf0f0f0, 139, 25);
_blueprintTitle.align
```

```
= TextFormatAlign.CENTER;
_blueprintTitle.text = _blueprintTitleText;

_blueprintBg.visible = false;
_blueprintFrame.visible = false;
_blueprintWeb.visible = false;
_blueprintName.visible = false;
_blueprintImage.visible = false;
_blueprintTitle.visible = false;

addChild(_bg);
addChild(_scrollbar);
addChild(_holder);

_holder.addChild(_victorBG);
_holder.addChild(_defeatedBG);
_holder.addChild(_victorsHeading);
_holder.addChild(_losersHeading);
_holder.addChild(_creditsFrame);
_holder.addChild(_alloyFrame);
_holder.addChild(_energyFrame);
_holder.addChild(_syntheticsFrame);
_holder.addChild(_quoteFrame);
_holder.addChild(_creditsCount);
_holder.addChild(_alloyCount);
_holder.addChild(_energyCount);
_holder.addChild(_syntheticsCount);
_holder.addChild(_quoteText);
_holder.addChild(_blueprintBg);
_holder.addChild(_blueprintFrame);
_holder.addChild(_blueprintWeb);
_holder.addChild(_blueprintImage);
_holder.addChild(_blueprintName);
_holder.addChild(_blueprintTitle);

presenter.addBattleLogDetailUpdatedListener(onBattleLogListUpdated);
presenter.addOnPlayerVOAddedListener(onPlayerVOUpdated);

if (_battleLog.hasDetails)
setUp(_battleLog);
else
presenter.getBattleLogDetails(_battleLog.battleKey);

addEffects();
effectsIN();
}

public function battleLog( v:BattleLogVO ):void
{
_battleLog = v;
//setUp(_battleLog);
```

```
}

private function onBattleLogListUpdated( v:BattleLogVO ):void
{
_battleLog = v;
setUp(_battleLog);
}

private function setUp( v:BattleLogVO ):void
{
var winners:Vector.<BattleLogPlayerInfoVO> = _battleLog.winners;
var losers:Vector.<BattleLogPlayerInfoVO> = _battleLog.losers;
var len:uint = winners.length;
var i:uint;
var creditsAmount:int;
var alloyAmount:int;
var energyAmount:int;
var syntheticsAmount:int;
var blueprint:String;
var bpAsset:AssetVO;
var currentPanel:BattleLogPlayerPanel;
var currentPlayer:BattleLogPlayerInfoVO;
var currentUserKey:String = CurrentUser.id;

if (len < 1)
_currentUsersOutcome = DRAW;

for (; i < len; ++i)
{
currentPlayer = winners[i];

if(currentPlayer == null)
continue;

currentPanel = new BattleLogPlayerPanel(currentPlayer, true);
currentPanel.setUp(presenter.getShipPrototypeByName, presenter.getAssetVOFromIPrototype,
presenter.loadPortraitMedium, presenter.loadPortraitIcon);
_holder.addChild(currentPanel);

if (_currentUsersOutcome != WINNER)
{
alloyAmount -= currentPlayer.alloyGained;
energyAmount -= currentPlayer.energyGained;
syntheticsAmount -= currentPlayer.syntheticGained;
}
if (currentPlayer.playerKey == currentUserKey)
{
_winners.unshift(currentPanel);
creditsAmount = currentPlayer.creditsGained;
alloyAmount
```

```
= currentPlayer.alloyGained;
energyAmount = currentPlayer.energyGained;
syntheticsAmount = currentPlayer.syntheticGained;
blueprint = currentPlayer.blueprintGained;
_currentUsersOutcome = WINNER;
} else
{
_winners.push(currentPanel);
currentPanel.onViewBaseClick.add(gotoBase);
presenter.requestPlayer(currentPlayer.playerKey);
}
}

len = losers.length;
for (i = 0; i < len; ++i)
{
currentPlayer = losers[i];
currentPanel = new BattleLogPlayerPanel(currentPlayer, false);
currentPanel.setUp(presenter.getShipPrototypeByName, presenter.getAssetVOFromIPrototype,
presenter.loadPortraitMedium, presenter.loadPortraitIcon);
_holder.addChild(currentPanel);

if (currentPlayer.playerKey == currentUserKey)
{
if (_currentUsersOutcome != DRAW)
_currentUsersOutcome = LOSER;
_losers.unshift(currentPanel);
blueprint = currentPlayer.blueprintGained;
} else
{
_losers.push(currentPanel);
currentPanel.onViewBaseClick.add(gotoBase);
presenter.requestPlayer(currentPlayer.playerKey);
}
}

if (blueprint != '')
_blueprint = presenter.getBlueprintPrototypeByName(blueprint);

if (_blueprint)
{
var rarity:String = _blueprint.getUnsafeValue('rarity');
if (rarity != 'Common')
{
var glow:GlowFilter = CommonFunctionUtil.getRarityGlow(rarity);
_blueprintName.textColor = glow.color;
_blueprintFrame.filters = [glow];
}

bpAsset = presenter.getAssetVOFromIPrototype(_blueprint);
_blueprintName.text
```

```
= bpAsset.visibleName;

_blueprintBg.visible = true
_blueprintFrame.visible = true;
_blueprintWeb.visible = true;
_blueprintName.visible = true;
_blueprintImage.visible = true;
_blueprintTitle.visible = true;
}


var resourceColor:uint;
var faction:String;
var victory:Boolean;
switch (_currentUsersOutcome)
{
case WINNER:
resourceColor = 0x7afe60;
faction = CurrentUser.faction;
victory = true;
break;
case LOSER:
resourceColor = 0xfe4f4f;
faction = BattleLogPlayerInfoVO(winners[Math.floor(CommonFunctionUtil.randomMinMax(0,
(winners.length - 1)))]).faction;
victory = false;
break;
case DRAW:
resourceColor = 0xfe4f4f;
faction = CurrentUser.faction;
victory = false;
break;
}

_syntheticsCount.textColor = _energyCount.textColor = _alloyCount.textColor =
_creditsCount.textColor = resourceColor;
_creditsCount.text = StringUtil.commaFormatNumber(creditsAmount);
_alloyCount.text = StringUtil.commaFormatNumber(alloyAmount);
_energyCount.text = StringUtil.commaFormatNumber(energyAmount);
_syntheticsCount.text = StringUtil.commaFormatNumber(syntheticsAmount);

var dialogueOptions:Vector.<IPrototype> = presenter.getBattleEndDialogByFaction(faction,
(victory) ? 'Victory' : 'Taunt');
_quoteText.text = dialogueOptions[Math.floor(CommonFunctionUtil.randomMinMax(0,
(dialogueOptions.length - 1)))].getValue('dialogString');

//todo uncomment when ready
//var audioDir:String = dialogueOptions[Math.floor(CommonFunctionUtil.randomMinMax(0,
(dialogueOptions.length - 1)))].getValue('dialogAudioString');
//if(audioDir.length>0)
//
```

```
presenter.playSound(audioDir, 0.75);

layout();

if (bpAsset)
presenter.loadIcon(bpAsset.mediumImage, layoutBlueprint);
}

private function onEntryClicked( entry:BattleLogEntry ):void
{
presenter.getBattleLogDetails(entry.battleKey);
}

private function onLoadImage( race:String, callback:Function ):void
{
if (presenter)
presenter.loadPortraitSmall(race, callback);
}

private function layoutBlueprint( asset:BitmapData ):void
{
_blueprintImage.onImageLoaded(asset);
_blueprintImage.x = _blueprintWeb.x + (_blueprintWeb.width - _blueprintImage.width) * 0.5;
_blueprintImage.y = _blueprintWeb.y + (_blueprintWeb.height - _blueprintImage.height) * 0.5;
}

private function layout():void
{
var hasBlueprint:Boolean = (_blueprint != null);


if (_currentUsersOutcome != DRAW)
layoutNormalConditions();
else
layoutDrawConditions();

var offset:int = (hasBlueprint) ? 41 : 10;
var yPos:Number = _defeatedBG.y + _defeatedBG.height + offset;

_creditsFrame.x = (hasBlueprint) ? 0 : 100;
_creditsFrame.y = yPos;

_alloyFrame.x = _creditsFrame.x + 175;
_alloyFrame.y = _creditsFrame.y;

_energyFrame.x = _creditsFrame.x;
_energyFrame.y = _creditsFrame.y + 50;

_syntheticsFrame.x = _alloyFrame.x;
_syntheticsFrame.y = _creditsFrame.y + 50;

_creditsCount.x
```

```
= _creditsFrame.x + 42;
_creditsCount.y = _creditsFrame.y + 16;

_alloyCount.x = _alloyFrame.x + 42;
_alloyCount.y = _alloyFrame.y + 16;

_energyCount.x = _energyFrame.x + 42;
_energyCount.y = _energyFrame.y + 16;

_syntheticsCount.x = _syntheticsFrame.x + 42;
_syntheticsCount.y = _syntheticsFrame.y + 16;

_blueprintBg.x = _alloyFrame.x + 173;
_blueprintBg.y = _defeatedBG.y + _defeatedBG.height + 3;

_blueprintFrame.x = _blueprintBg.x + (_blueprintBg.width - _blueprintFrame.width) * 0.5;
_blueprintFrame.y = _blueprintBg.y + (_blueprintBg.height - _blueprintFrame.height) * 0.5 - 8;

_blueprintWeb.x = _blueprintFrame.x + (_blueprintFrame.width - _blueprintWeb.width) * 0.5;
_blueprintWeb.y = _blueprintFrame.y + (_blueprintFrame.height - _blueprintWeb.height) * 0.5;

_blueprintName.x = _blueprintBg.x + 1;
_blueprintName.y = _blueprintBg.y + _blueprintBg.height - _blueprintName.textHeight - 6;

_blueprintTitle.x = _blueprintBg.x - 1;
_blueprintTitle.y = _blueprintBg.y + _blueprintBg.height - _blueprintTitle.textHeight + 20;
_blueprintTitle.rotation = -90;

_quoteFrame.y = (hasBlueprint) ? (_blueprintBg.y + _blueprintBg.height + 10) :
(_syntheticsFrame.y + _syntheticsFrame.height + 38);

_quoteText.x = _quoteFrame.x + 11;
_quoteText.y = _quoteFrame.y + 5;

_maxHeight = _quoteFrame.y + _quoteFrame.height + 8;

_scrollbar.updateScrollableHeight(_maxHeight);
}

private function layoutNormalConditions():void
{
_losersHeading.text = _defeatedText;

_victorsHeading.x = 12;
_victorsHeading.y = -4;
_victorBG.y = 22;

var i:uint;
var len:uint = _winners.length;
var selection:BattleLogPlayerPanel;
var
```

```
xPos:int = 11;
var yPos:int = _victorBG.y + 5;
_maxHeight = 0;
for (i = 0; i < len; ++i)
{
selection = _winners[i];
selection.y = yPos;
yPos += selection.height + 10;
}

if (yPos > _victorBG.y + _victorBG.height)
_victorBG.height = yPos - 2;

_defeatedBG.y = _victorBG.y + _victorBG.height + 26;

_losersHeading.x = 5;
_losersHeading.y = _defeatedBG.y - _losersHeading.textHeight - 2;

yPos = _defeatedBG.y + 5;
len = _losers.length;
for (i = 0; i < len; ++i)
{
selection = _losers[i];
selection.y = yPos;
yPos += selection.height + 10;
}

if (yPos > _defeatedBG.y + _defeatedBG.height)
_defeatedBG.height = (yPos - _defeatedBG.y) + 20;
}

private function layoutDrawConditions():void
{

var i:uint;
var len:uint = _winners.length;
var selection:BattleLogPlayerPanel;
var xPos:int = 11;
var yPos:int = _victorBG.y;

_losersHeading.text = _drawText;

_victorsHeading.visible = false;
_victorBG.visible = false;

_losersHeading.x = 12;
_losersHeading.y = -4;
_defeatedBG.y = 22;

yPos = _defeatedBG.y + 5;
len
```

```
= _losers.length;
for (i = 0; i < len; ++i)
{
selection = _losers[i];
selection.y = yPos;
yPos += selection.height + 10;
}

if (yPos > _defeatedBG.y + _defeatedBG.height)
_defeatedBG.height = (yPos - _defeatedBG.y) + 20;
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

private function onPlayerVOUpdated( player:PlayerVO ):void
{
var i:uint;
var len:uint = _winners.length;
var selection:BattleLogPlayerPanel;
var playerUpdated:Boolean;
for (i = 0; i < len; ++i)
{
selection = _winners[i];
if (selection.playerID == player.id)
{
playerUpdated = true;
selection.player = player;
break;
}

}

if (!playerUpdated)
{
len = _losers.length;
for (i = 0; i < len; ++i)
{
selection = _losers[i];
if (selection.playerID == player.id)
{
playerUpdated = true;
selection.player = player;
break;
}
}
}
}
```

```
private function gotoBase( baseXPos:Number, baseYPos:Number, baseSector:String ):void
{
presenter.gotoCoords(baseXPos, baseYPos, baseSector);
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
presenter.removeBattleLogDetailUpdatedListener(onBattleLogListUpdated);
presenter.removeOnPlayerVOAddedListener(onPlayerVOUpdated);
super.destroy();

_bg = null;
_quoteFrame = null;

_holder = null;

_scrollbar.destroy();
_scrollbar = null;

var len:uint = _winners.length;
var i:uint;
var currentPlayer:BattleLogPlayerPanel;
for (; i < len; ++i)
{
currentPlayer = _winners[i];
currentPlayer.destroy();
currentPlayer = null;
}
_winners.length = 0;

len = _losers.length;
for (i = 0; i < len; ++i)
{
currentPlayer = _losers[i];
currentPlayer.destroy();
currentPlayer = null;
}
_losers.length = 0;

_scrollRect = null;

_holder
```

```
= null;

_battleLog = null;

_alloyCount.destroy();
_alloyCount = null;

_creditsCount.destroy();
_creditsCount = null;

_energyCount.destroy();
_energyCount = null;

_syntheticsCount.destroy();
_syntheticsCount = null;

_blueprintTitle.destroy();
_blueprintTitle = null;

_blueprintName.destroy();
_blueprintName = null;

_quoteText.destroy();
_quoteText = null;

_victorsHeading.destroy();
_victorsHeading = null;

_losersHeading.destroy();
_losersHeading = null;

_blueprintFrame = null;
_blueprintWeb = null;
_blueprintBg = null;
_alloyFrame = null;
_creditsFrame = null;
_energyFrame = null;
_syntheticsFrame = null;

ObjectPool.give(_blueprintImage);
_blueprintImage = null;

_victorBG = null;
_defeatedBG = null;

_currentUserHasWon = false;
_blueprint = null;

_maxHeight = 0;
}

}
```

```
}

-----------------------------------
File 793: igw\com\ui\modal\battlelog\BattleLogEntry.as
package com.ui.modal.battlelog
{
import com.enum.ui.ButtonEnum;
import com.model.battlelog.BattleLogBaseInfoVO;
import com.model.battlelog.BattleLogPlayerInfoVO;
import com.model.battlelog.BattleLogVO;
import com.model.player.CurrentUser;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.events.MouseEvent;
import flash.geom.Point;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;
import flash.utils.getTimer;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class BattleLogEntry extends BitmapButton
{
private var _bg:Bitmap;

private var _currentUserImage:ImageComponent;
private var _currentUserPortraitFrame:Bitmap;
private var _currentUserName:Label;
private var _currentUserOutcome:Label;
private var _currentUserRating:Label;

private var _enemyPlayerImage:ImageComponent;
private var _enemyPlayerPortraitFrame:Bitmap;
private var _enemyPlayerName:Label;
private var _enemyPlayerOutcome:Label;
private var _enemyRating:Label;

private var _viewReplayBtn:BitmapButton;

private
```

```
var _battleOutcome:Label;
private var _timeSinceBattle:Label;

private var _battleLog:BattleLogVO;

public var onClicked:Signal;
public var onReplayClicked:Signal;
public var onLoadImage:Signal;

private var _victory:String = 'CodeString.BattleLog.Victory';
private var _defeat:String = 'CodeString.BattleLog.Defeat';
private var _draw:String = 'CodeString.BattleLog.Draw';
private var _victoryParens:String = 'CodeString.BattleLog.VictoryParens';
private var _defeatParens:String = 'CodeString.BattleLog.DefeatParens';
private var _drawParens:String = 'CodeString.BattleLog.DrawParens';
private var _fleetRatingText:String = 'CodeString.BattleLogs.FleetRating';
private var _baseRatingText:String = 'CodeString.BattleLogs.BaseRating';
private var _timeSinceBattleText:String = 'CodeString.BattleLog.TimeSinceBattle';
private var _viewReplayText:String = 'CodeString.BattleLog.ViewReplay'; //VIEW REPLAY

public function BattleLogEntry( battleLog:BattleLogVO )
{
onClicked = new Signal(BattleLogVO);
onReplayClicked = new Signal(BattleLogVO);
onLoadImage = new Signal(String, Function);

_battleLog = battleLog;
var windowBGClass:Class = Class(getDefinitionByName('BattleLogContainerBMD'));

_currentUserImage = ObjectPool.get(ImageComponent);
_currentUserImage.init(2000, 2000);
_currentUserImage.x = 6;
_currentUserImage.y = 9;

_currentUserPortraitFrame = new Bitmap();
_currentUserPortraitFrame.x = 2;
_currentUserPortraitFrame.y = 6;

_currentUserName = new Label(24, 0xf0f0f0, 300, 25, false);
_currentUserName.x = 95;
_currentUserName.y = 15;
_currentUserName.align = TextFormatAlign.LEFT;

_currentUserRating = new Label(20, 0xfbefaf, 300, 25);
_currentUserRating.x = 95;
_currentUserRating.y = 38;
_currentUserRating.align = TextFormatAlign.LEFT;

_currentUserOutcome = new Label(20, 0xf0f0f0, 300, 25);
_currentUserOutcome.x = 95;
_currentUserOutcome.y
```

```
= 64;
_currentUserOutcome.align = TextFormatAlign.LEFT;

var replayButtonWidth:int = 130;

_enemyPlayerImage = ObjectPool.get(ImageComponent);
_enemyPlayerImage.init(2000, 2000);
_enemyPlayerImage.x = 601-replayButtonWidth;
_enemyPlayerImage.y = 9;

_enemyPlayerPortraitFrame = new Bitmap();
_enemyPlayerPortraitFrame.x = 597-replayButtonWidth;
_enemyPlayerPortraitFrame.y = 6;

_enemyPlayerName = new Label(24, 0xf0f0f0, 300, 25, false);
_enemyPlayerName.x = 289-replayButtonWidth;
_enemyPlayerName.y = 15;
_enemyPlayerName.align = TextFormatAlign.RIGHT;

_enemyRating = new Label(20, 0xfbefaf, 300, 25);
_enemyRating.x = 289-replayButtonWidth;
_enemyRating.y = 38;
_enemyRating.align = TextFormatAlign.RIGHT;

_enemyPlayerOutcome = new Label(20, 0xf0f0f0, 300, 25);
_enemyPlayerOutcome.x = 289-replayButtonWidth;
_enemyPlayerOutcome.y = 64;
_enemyPlayerOutcome.align = TextFormatAlign.RIGHT;

super.init(BitmapData(new windowBGClass()));

if( battleLog.hasReplay )
{
_viewReplayBtn = UIFactory.getButton(ButtonEnum.BLUE_A, replayButtonWidth-8, 50,
_bitmap.width - replayButtonWidth, 20, _viewReplayText);
_viewReplayBtn.visible = true;
_viewReplayBtn.addEventListener(MouseEvent.MOUSE_UP, onViewReplayButtonClick, false,
0, true);
}

_battleOutcome = new Label(40, 0xf0f0f0);
_battleOutcome.constrictTextToSize = false;
_battleOutcome.autoSize = TextFieldAutoSize.CENTER;
_battleOutcome.x = _bitmap.x + _bitmap.width * 0.5 - replayButtonWidth * 0.5;

_timeSinceBattle = new Label(16, 0xfbefaf, 200);
_timeSinceBattle.align = TextFormatAlign.LEFT;
_timeSinceBattle.constrictTextToSize = false;
_timeSinceBattle.x = 10;
_timeSinceBattle.y -= _timeSinceBattle.height;

addChild(_currentUserImage);
```

```
addChild(_currentUserPortraitFrame);
addChild(_currentUserName);
addChild(_currentUserRating);
addChild(_currentUserOutcome);

addChild(_enemyPlayerImage);
addChild(_enemyPlayerPortraitFrame);
addChild(_enemyPlayerName);
addChild(_enemyRating);
addChild(_enemyPlayerOutcome);

if( _viewReplayBtn )
{
addChild( _viewReplayBtn );
}

addChild(_battleOutcome);
addChild(_timeSinceBattle);
}

public function setUp():void
{
var i:uint;
var currentPlayer:BattleLogPlayerInfoVO;
var winners:Vector.<BattleLogPlayerInfoVO> = _battleLog.winners;
var losers:Vector.<BattleLogPlayerInfoVO> = _battleLog.losers;
var len:uint = winners.length;
var enemyPlayer:BattleLogPlayerInfoVO;
var currentUser:BattleLogPlayerInfoVO;
var currentUsersIndex:int;
var currentUserInWinners:Boolean;
for (; i < len; ++i)
{
currentPlayer = winners[i];
if (currentPlayer.playerKey == CurrentUser.id)
{
currentUserInWinners = true;
currentUser = currentPlayer;
currentUsersIndex = i;
break;
}

}

if (currentUser == null)
{
len = losers.length;
for (i = 0; i < len; ++i)
{
currentPlayer
```

```
= losers[i];
if (currentPlayer.playerKey == CurrentUser.id)
{
currentUser = currentPlayer;
currentUsersIndex = i;
break;
}
}
}

if( currentUser == null && winners.length)
{
currentUser = winners[0];
currentUserInWinners = true;
}
else if( currentUser == null && losers.length)
{
currentUser = losers[0];
currentUserInWinners = false;
}

if (currentUserInWinners)
{
enemyPlayer = losers[0];
_currentUserOutcome.text = _victoryParens;
_battleOutcome.text = _victory;
_battleOutcome.textColor = _currentUserOutcome.textColor = 0xffdd3d;
_enemyPlayerOutcome.text = _defeatParens;
_enemyPlayerOutcome.textColor = 0xec1f1f;
} else if (!currentUserInWinners && winners.length < 1)
{
enemyPlayer = (currentUsersIndex + 1 > (losers.length - 1)) ? losers[0] :
losers[currentUsersIndex + 1];
_currentUserOutcome.text = _drawParens;
_battleOutcome.text = _draw;
_battleOutcome.textColor = _currentUserOutcome.textColor = 0xec1f1f;
_enemyPlayerOutcome.text = _drawParens;
_enemyPlayerOutcome.textColor = 0xec1f1f;
} else
{
enemyPlayer = winners[0];
_currentUserOutcome.text = _defeatParens;
_battleOutcome.text = _defeat;
_battleOutcome.textColor = _currentUserOutcome.textColor = 0xec1f1f;
_enemyPlayerOutcome.text = _victoryParens;
_enemyPlayerOutcome.textColor = 0xffdd3d;
}

_timeSinceBattle.setBuildTime(_battleLog.timeSince / 1000, 3);
_timeSinceBattle.setTextWithTokens(_timeSinceBattleText,
{'[[String.TimeSinceLastBattle]]':_timeSinceBattle.text});
```

```
var currentUserFactionColor:uint = CommonFunctionUtil.getFactionColor(currentUser.faction);
var enemyUserFactionColor:uint = CommonFunctionUtil.getFactionColor(enemyPlayer.faction);

var portraitFrameClass:Class =
Class(getDefinitionByName('BattleLogSmallPortraitFrameBMD'));

var playerBMD:BitmapData = BitmapData(new portraitFrameClass());
playerBMD.applyFilter(playerBMD, playerBMD.rect, new Point(0, 0),
CommonFunctionUtil.getColorMatrixFilter(currentUserFactionColor));

var enemyBMD:BitmapData = BitmapData(new portraitFrameClass());
enemyBMD.applyFilter(enemyBMD, enemyBMD.rect, new Point(0, 0),
CommonFunctionUtil.getColorMatrixFilter(enemyUserFactionColor));

onLoadImage.dispatch(currentUser.race, _currentUserImage.onImageLoaded);
var rating:String = (currentUser.wasBase) ? _baseRatingText : _fleetRatingText;
_currentUserName.text = currentUser.name;
_currentUserName.textColor = currentUserFactionColor;
_currentUserPortraitFrame.bitmapData = playerBMD;
_currentUserRating.setTextWithTokens(rating, {'[[Number.Rating]]':currentUser.rating});


onLoadImage.dispatch(enemyPlayer.race, _enemyPlayerImage.onImageLoaded);
rating = (enemyPlayer.wasBase) ? _baseRatingText : _fleetRatingText;
_enemyPlayerName.text = enemyPlayer.name;
_enemyPlayerName.textColor = enemyUserFactionColor;
_enemyPlayerPortraitFrame.bitmapData = enemyBMD;
_enemyRating.setTextWithTokens(rating, {'[[Number.Rating]]':enemyPlayer.rating});


_battleOutcome.y = _bitmap.y + (_bitmap.height - _battleOutcome.textHeight) * 0.5;
}

public function get timeOccurred():Number { return _battleLog.timeSince; }
public function get battleKey():String { return _battleLog.battleKey; }

override protected function onMouse( e:MouseEvent ):void
{
super.onMouse(e);
if (mouseEnabled)
{
switch (e.type)
{
case MouseEvent.CLICK:
onClicked.dispatch(_battleLog);
break;
}
}
}
```

```
private function onViewReplayButtonClick( e:MouseEvent ):void
{
onReplayClicked.dispatch( _battleLog );
}

override public function destroy():void
{
super.destroy();

onClicked.removeAll();
onClicked = null;

onReplayClicked.removeAll();
onReplayClicked = null;

onLoadImage.removeAll();
onLoadImage = null;

_bg = null;

ObjectPool.give(_currentUserImage);

_currentUserPortraitFrame = null;

_currentUserName.destroy();
_currentUserName = null;

_currentUserOutcome.destroy();
_currentUserOutcome = null;

_currentUserRating.destroy();
_currentUserRating = null;


ObjectPool.give(_enemyPlayerImage);

_enemyPlayerPortraitFrame = null;

_enemyPlayerName.destroy();
_enemyPlayerName = null;

_enemyPlayerOutcome.destroy();
_enemyPlayerOutcome = null;

_enemyRating.destroy();
_enemyRating = null;

_battleOutcome.destroy();
_battleOutcome
```

```
= null;

_timeSinceBattle.destroy();
_timeSinceBattle = null;

if (_viewReplayBtn)
{
_viewReplayBtn.destroy();
}
_viewReplayBtn = null;


_battleLog = null;
}
}
}

------------------------------------
File 794: igw\com\ui\modal\battlelog\BattleLogListView.as
package com.ui.modal.battlelog
{
import com.Application;
import com.enum.BattleLogFilterEnum;
import com.model.battlelog.BattleLogVO;
import com.presenter.shared.IUIPresenter;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.accordian.AccordianComponent;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

import org.shared.ObjectPool;

public class BattleLogListView extends View
{
private var _bg:DefaultWindowBG;
private var _battleLogEntries:Dictionary;
private var _noBattleLogs:Label;
private
```

```
var _accordian:AccordianComponent;

private var _battleLogs:Vector.<BattleLogEntry>;

private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _holder:Sprite;

private var _titleText:String = 'CodeString.BattleLogs.Title';
private var _noBattleLogsText:String = 'CodeString.BattleLog.NoBattleLogs'; //Your faction
needs your strength - don't be shy, enter the fray!

// accordian entries for results filtering
private var _filterSelfAllText:String = 'CodeString.BattleLog.ShowSelfAll';
private var _filterSelfPvPText:String = 'CodeString.BattleLog.ShowSelfPvP';
private var _filterSelfPvEText:String = 'CodeString.BattleLog.ShowSelfPvE';
private var _filterFleetPvPText:String = 'CodeString.BattleLog.ShowFleetPvP';
private var _filterBasePvPText:String = 'CodeString.BattleLog.ShowBasePvP';
private var _filterBestPvEText:String = 'CodeString.BattleLog.ShowBestPvE';

[PostConstruct]
override public function init():void
{
super.init();
_battleLogEntries = new Dictionary;
_battleLogs = new Vector.<BattleLogEntry>;

var filterWidth:int = ( Application.BATTLE_WEB_PATH )?150:0;

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(722+filterWidth, 475);
_bg.addTitle(_titleText, 239);
_bg.x -= 21;

if( Application.BATTLE_WEB_PATH )
{
_accordian = ObjectPool.get(AccordianComponent);
_accordian.init(140, 52);
_accordian.x = _bg.bg.x;
_accordian.y = _bg.bg.y + 5;
_accordian.addListener(onAccordianSelected);
_accordian.addGroup(BattleLogFilterEnum.SELFALL, _filterSelfAllText);
_accordian.addGroup(BattleLogFilterEnum.SELFPVP, _filterSelfPvPText);
_accordian.addGroup(BattleLogFilterEnum.SELFPVE, _filterSelfPvEText);
_accordian.addGroup(BattleLogFilterEnum.BASEPVP, _filterBasePvPText);
_accordian.addGroup(BattleLogFilterEnum.FLEETPVP, _filterFleetPvPText);
_accordian.addGroup(BattleLogFilterEnum.BESTPVE, _filterBestPvEText);
}
```

```
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);
addChild(_bg);

_noBattleLogs = new Label(24, 0xf0f0f0);
_noBattleLogs.constrictTextToSize = false;
_noBattleLogs.autoSize = TextFieldAutoSize.CENTER;
_noBattleLogs.x = _bg.x + (_bg.width - _noBattleLogs.textWidth) * 0.5;
_noBattleLogs.y = _bg.y + (_bg.height - _noBattleLogs.textHeight) * 0.5;
_noBattleLogs.text = _noBattleLogsText;

_holder = new Sprite();
_holder.x = 6+filterWidth;
_holder.y = 50;
_maxHeight = 0;

_scrollRect = new Rectangle(0, 0, 682+filterWidth, 448);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect;

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = 690+filterWidth;
var scrollbarYPos:Number = 78;
_scrollbar.init(7, _scrollRect.height - 10, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 28.25;

addChild(_bg);
if( _accordian )
{
addChild(_accordian);
}
addChild(_noBattleLogs);
addChild(_scrollbar);
addChild(_holder);

presenter.addBattleLogListUpdatedListener(onBattleLogListUpdated);
presenter.getBattleLogList( BattleLogFilterEnum.SELFALL);
if( _accordian )
{
_accordian.setSelected(BattleLogFilterEnum.SELFALL, "");
}
addEffects();
effectsIN();
}

private
```

```
function onBattleLogListUpdated( v:Vector.<BattleLogVO> ):void
{
destroyEntries();
_battleLogEntries = new Dictionary;

var len:int = v.length;
var currentBattleLogVO:BattleLogVO;
var currentBattleLogEntry:BattleLogEntry;
for (var i:int = 0; i < len; ++i)
{
currentBattleLogVO = v[i];
if (currentBattleLogVO.battleKey in _battleLogEntries)
currentBattleLogEntry = _battleLogEntries[currentBattleLogVO.battleKey];
else
{
currentBattleLogEntry = new BattleLogEntry(currentBattleLogVO);
currentBattleLogEntry.onClicked.add(onEntryClicked);
currentBattleLogEntry.onLoadImage.add(onLoadImage);
currentBattleLogEntry.onReplayClicked.add(onReplay);
currentBattleLogEntry.setUp();
_battleLogs.push(currentBattleLogEntry);
}
_battleLogEntries[currentBattleLogVO.battleKey] = currentBattleLogEntry;
_holder.addChild(currentBattleLogEntry);
}

_battleLogs.sort(orderItems);
layout();
}

private function onEntryClicked( log:BattleLogVO ):void
{
var battleLogDetailView:BattleLogDetailView =
BattleLogDetailView(_viewFactory.createView(BattleLogDetailView));
battleLogDetailView.battleLog(log);
_viewFactory.notify(battleLogDetailView);
}

private function onReplay( log:BattleLogVO ):void
{
var battleId:String = log.battleKey.replace(".battleLog", "");
presenter.viewBattleReplay( battleId );
}

private function onAccordianSelected( groupID:String, subItemID:String, data:* ):void
{
presenter.getBattleLogList( groupID );
}

private function onLoadImage( race:String, callback:Function ):void
{
```

```
if (presenter)
presenter.loadPortraitSmall(race, callback);
}

protected function layout():void
{
var len:uint = _battleLogs.length;
var selection:BattleLogEntry;
var yPos:int = 20;
_noBattleLogs.visible = (len > 0) ? false : true;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _battleLogs[i];
selection.y = yPos;
_maxHeight += selection.height;
yPos += selection.height;
}
_scrollbar.updateScrollableHeight(_maxHeight);
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

private function orderItems( itemOne:BattleLogEntry, itemTwo:BattleLogEntry ):Number
{

if (!itemOne)
return -1;
if (!itemTwo)
return 1;

var timeOccurredOne:Number = itemOne.timeOccurred;
var timeOccurredTwo:Number = itemTwo.timeOccurred;

if (timeOccurredOne < timeOccurredTwo)
return -1;
else if (timeOccurredOne > timeOccurredTwo)
return 1;

return 0;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
```

```
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

private function destroyEntries():void
{
var len:uint = _battleLogs.length;
var battleLogEntry:BattleLogEntry;
for (var i:uint = 0; i < len; ++i)
{
battleLogEntry = _battleLogs[i];
_holder.removeChild(battleLogEntry);
battleLogEntry.destroy();
battleLogEntry = null;
}
_battleLogs.length = 0;

for (var key:String in _battleLogEntries)
{
delete _battleLogEntries[key];
}
_battleLogEntries = null;
}

override public function destroy():void
{
presenter.removeBattleLogListUpdatedListener(onBattleLogListUpdated)
super.destroy();

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if( _accordian )
{
ObjectPool.give(_accordian);
_accordian = null;
}

destroyEntries();


_holder = null;

_scrollbar.destroy();
_scrollbar = null;

_noBattleLogs.destroy();
_noBattleLogs = null;

_maxHeight
```

```
= 0;
}
}
}

------------------------------------
File 795: igw\com\ui\modal\battlelog\BattleLogPlayerPanel.as
package com.ui.modal.battlelog
{
import com.enum.ui.ButtonEnum;
import com.model.asset.AssetVO;
import com.model.battlelog.BattleLogEntityInfoVO;
import com.model.battlelog.BattleLogPlayerInfoVO;
import com.model.player.PlayerVO;
import com.model.prototype.IPrototype;
import com.service.server.incoming.data.BattleLogEntityDetailInfo;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.ui.modal.dock.ShipIcon;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.ColorTransform;
import flash.geom.Point;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class BattleLogPlayerPanel extends Sprite
{
public var onViewBaseClick:Signal;

private var _playerPortrait:ImageComponent;

private var _playerPortraitFrame:Bitmap;
private var _shipsInFleetBg:Bitmap;
private var _fleetFrame:Bitmap;

private var _playerName:Label;
private var _playerRating:Label;

private
```

```
var _entityText:Vector.<Label>;
private var _gotoBaseBtn:BitmapButton;

private var _isWinner:Boolean;

private var _ships:Vector.<ShipIcon>;

private var _battleLogPlayer:BattleLogPlayerInfoVO;
private var _player:PlayerVO;

private var _getShipPrototype:Function;
private var _getUIAsset:Function;
private var _loadPortraitMedium:Function;
private var _loadPortraitIcon:Function;

private var WINNER_PORTRAIT_X:Number = 5;
private var LOSER_PORTRAIT_X:Number = 391;

private var WINNER_GOTO_BASE_X:Number = 244;
private var LOSER_GOTO_BASE_X:Number = 167;

private var WINNER_BATTLE_TEXT_X:Number = 139;
private var LOSER_BATTLE_TEXT_X:Number = 174;

private var BATTLE_ENTITY_Y:Number = 58;

private var WINNER_BATTLE_SHIP_X:Number = 361;
private var LOSER_BATTLE_SHIP_X:Number = 25;
private var BATTLE_SHIP_Y:Number = 8;

private var PANEL_HEIGHT:Number = 143;

private var _leftEntityText:String = 'CodeString.BattleLogs.EntityHealthLeft';
private var _RightEntityText:String = 'CodeString.BattleLogs.EntityHealthRight';
private var _fleetRatingText:String = 'CodeString.BattleLogs.FleetRating';
private var _baseRatingText:String = 'CodeString.BattleLogs.BaseRating';
private var _baseHealthText:String = 'CodeString.BattleLog.BaseHealth';
private var _viewBaseText:String = 'CodeString.Shared.ViewBase' //VIEW BASE

public function BattleLogPlayerPanel( battleLogPlayer:BattleLogPlayerInfoVO, won:Boolean )
{
onViewBaseClick = new Signal(Number, Number, String);

_ships = new Vector.<ShipIcon>;
_entityText = new Vector.<Label>;
_battleLogPlayer = battleLogPlayer;
_isWinner = won;
var factionColor:uint = CommonFunctionUtil.getFactionColor(_battleLogPlayer.faction);
var ratingText:String = (_battleLogPlayer.hasFleet) ? _fleetRatingText : _baseRatingText;
var ratingVariables:Object = (_battleLogPlayer.hasFleet) ?
{'[[Number.Rating]]':_battleLogPlayer.fleet.fleetRating()}
```

```
: {'[[Number.Rating]]':_battleLogPlayer.base.baseRatings};

var portraitFrameClass:Class =
Class(getDefinitionByName('BattleLogLargePortraitFrameBMD'));
_playerPortrait = ObjectPool.get(ImageComponent);
_playerPortrait.init(2000, 2000);

var bmd:BitmapData = BitmapData(new portraitFrameClass());
bmd.applyFilter(bmd, bmd.rect, new Point(0, 0),
CommonFunctionUtil.getColorMatrixFilter(factionColor));
_playerPortraitFrame = new Bitmap(bmd);

_playerName = new Label(24, 0xf0f0f0, 210, 25);
_playerName.constrictTextToSize = false;
_playerName.align = (_isWinner) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
_playerName.text = _battleLogPlayer.name;
_playerName.textColor = factionColor;

_playerRating = new Label(20, 0xfbefaf, 210, 25);
_playerRating.constrictTextToSize = false;
_playerRating.align = (_isWinner) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
_playerRating.constrictTextToSize = false;
_playerRating.setTextWithTokens(ratingText, ratingVariables);

_gotoBaseBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 130, 29, 0, 0, _viewBaseText);
_gotoBaseBtn.visible = false;
_gotoBaseBtn.addEventListener(MouseEvent.MOUSE_UP, onButtonClick, false, 0, true);

_fleetFrame = UIFactory.getBitmap('SectorFleetSelectionBGBMD');


addChild(_playerPortrait);
addChild(_playerName);
addChild(_playerRating);
addChild(_playerPortraitFrame);
addChild(_fleetFrame);
addChild(_gotoBaseBtn);
}

public function setUp( getShipPrototype:Function, getUIAsset:Function,
loadPortraitMedium:Function, loadPortraitIcon:Function ):void
{
_getShipPrototype = getShipPrototype;
_getUIAsset = getUIAsset;
_loadPortraitMedium = loadPortraitMedium;
_loadPortraitIcon = loadPortraitIcon;

if (_battleLogPlayer.hasFleet)
setUpPlayerFleet();
else
setUpBase();
```

```
_loadPortraitMedium(_battleLogPlayer.race, _playerPortrait.onImageLoaded);
layout();
}

private function layout():void
{

_playerName.x = (_isWinner) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
_playerName.y = 8;

_playerRating.x = (_isWinner) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
_playerRating.y = 35;

_playerPortrait.x = ((_isWinner) ? WINNER_PORTRAIT_X : LOSER_PORTRAIT_X) + 5;
_playerPortrait.y = 8;

_playerPortraitFrame.x = ((_isWinner) ? WINNER_PORTRAIT_X : LOSER_PORTRAIT_X);
_playerPortraitFrame.y = 3;

_gotoBaseBtn.x = _playerPortraitFrame.x + (_playerPortraitFrame.width - _gotoBaseBtn.width) *
0.5;
_gotoBaseBtn.y = _playerPortraitFrame.y + _playerPortraitFrame.height + 2;

_fleetFrame.x = (_isWinner) ? WINNER_BATTLE_SHIP_X - 1 : LOSER_BATTLE_SHIP_X - 1;
_fleetFrame.y = BATTLE_SHIP_Y - 1;

var xPos:Number = (_isWinner) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
var yPos:Number = BATTLE_ENTITY_Y;
var len:uint = _entityText.length;
var currentLabel:Label;
for (var i:uint = 0; i < len; ++i)
{
currentLabel = _entityText[i];
currentLabel.x = xPos;
currentLabel.y = yPos;

yPos += currentLabel.textHeight;
}
}

private function setUpBase():void
{
_fleetFrame.visible = false;

var healthValue:Number = Math.round(_battleLogPlayer.base.health * 100);
var healthColor:uint = getHealthColor(healthValue);
var textToUse:String = (_isWinner) ? _leftEntityText : _RightEntityText;
var base:Label = new Label(12, 0xf0f0f0, 210, 25);
base.align
```

```
= (_isWinner) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
base.constrictTextToSize = false;
base.x = (_isWinner) ? WINNER_BATTLE_TEXT_X : LOSER_BATTLE_TEXT_X;
base.y = BATTLE_ENTITY_Y;
base.setHtmlTextWithTokens(textToUse, {'[[HexNumber.Color]]':healthColor.toString(16),
'[[Number.EntityHealth]]':healthValue, '[[String.EntityName]]':_baseHealthText});
_entityText.push(base);
addChild(base);
}

private function setUpPlayerFleet():void
{
_fleetFrame.visible = true;

var image:ShipIcon;
var xPos:Number;
var yPos:Number;

var ships:Vector.<BattleLogEntityInfoVO> = _battleLogPlayer.fleet.ships
var currentShip:BattleLogEntityDetailInfo;
var shipProto:IPrototype;
for (var i:uint = 0; i < 6; ++i)
{
if( ships.length > i )
{
shipProto = _getShipPrototype(ships[i].protoName);
}
else
{
shipProto = null;
}
image = new ShipIcon();
image.scale(0.37, 0.37);

xPos = (_isWinner) ? WINNER_BATTLE_SHIP_X : LOSER_BATTLE_SHIP_X;
yPos = BATTLE_SHIP_Y;

switch (i)
{
case 0:
xPos += 46;
yPos -= 1;
break;
case 1:
xPos += 0;
yPos += 24;
break;
case 2:
xPos += 90;
yPos += 24;
break;
```

```
case 3:
xPos += 0;
yPos += 76;
break;
case 4:
xPos += 90;
yPos += 76;
break;
case 5:
xPos += 45;
yPos += 103;
break;
}

if (shipProto)
{
image.onLoadShipImage.add(_loadPortraitIcon);
image.setShip(null, shipProto);
image.setBarValue(1 - ships[i].health);
addShipLabel(shipProto, ships[i].health);
}

image.x = xPos;
image.y = yPos;
image.mouseEnabled = false;
addChild(image);
_ships.push(image);
}

}

private function addShipLabel( shipProto:IPrototype, health:Number ):void
{
var healthValue:Number = Math.round(health * 100);
var shipUIAsset:AssetVO = _getUIAsset(shipProto);
var healthColor:uint = getHealthColor(healthValue);
var textToUse:String = (_isWinner) ? _leftEntityText : _RightEntityText;
var ship:Label = new Label(14, 0xf0f0f0, 210, 25);
ship.constrictTextToSize = false;
ship.align = (_isWinner) ? TextFormatAlign.LEFT : TextFormatAlign.RIGHT;
ship.setHtmlTextWithTokens(textToUse, {'[[HexNumber.Color]]':healthColor.toString(16),
'[[Number.EntityHealth]]':healthValue, '[[String.EntityName]]':shipUIAsset.visibleName});
_entityText.push(ship);
addChild(ship);

}

private function getHealthColor( healthValue:Number ):uint
{

if
```

```
(healthValue >= 75)
return 0x3cf219;
else if (healthValue >= 50)
return 0xfbe81a;
else if (healthValue >= 25)
return 0xfa7d0e;
else
return 0xf81919;
}

private function onButtonClick( e:MouseEvent ):void
{
if (_player)
{
onViewBaseClick.dispatch(_player.baseXPos, _player.baseYPos, _player.baseSector);
}
}

public function get playerID():String
{
return _battleLogPlayer.playerKey;
}

public function set player( player:PlayerVO ):void
{
_player = player;
_gotoBaseBtn.visible = true;
}

override public function get height():Number
{
return PANEL_HEIGHT;
}

public function destroy():void
{
if (onViewBaseClick)
onViewBaseClick.removeAll();
onViewBaseClick = null;

ObjectPool.give(_playerPortrait);
_playerPortrait = null;

_playerName.destroy();
_playerName = null;

_playerRating.destroy();
_playerRating = null;

_playerPortraitFrame = null;

if
```

```
(_gotoBaseBtn)
{
_gotoBaseBtn.removeEventListener(MouseEvent.MOUSE_UP, onButtonClick);
_gotoBaseBtn.destroy();
}
_gotoBaseBtn = null;

var len:uint = _ships.length;
var i:uint;
var currentShipIcon:ShipIcon;
for (; i < len; ++i)
{
currentShipIcon = _ships[i];
currentShipIcon.destroy();
currentShipIcon = null;
}
_ships.length = 0;
len = _entityText.length;
var currentLabel:Label;
for (i = 0; i < len; ++i)
{
currentLabel = _entityText[i];
currentLabel.destroy();
currentLabel = null;
}

_isWinner = false;

_battleLogPlayer = null;

_getShipPrototype = null;
_getUIAsset = null;
_loadPortraitMedium = null;
_loadPortraitIcon = null;
}
}
}

------------------------------------
File 796: igw\com\ui\modal\building\RefitBuildingView.as
package com.ui.modal.building
{
import com.controller.transaction.requirements.PurchaseVO;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.CurrencyEnum;
import com.enum.SlotComponentEnum;
import com.enum.TypeEnum;
import com.enum.server.PurchaseTypeEnum;
import com.enum.ui.ButtonEnum;
import com.event.TransactionEvent;
import
```

```
com.model.asset.AssetModel;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.IStarbasePresenter;
import com.service.language.Localization;
import com.ui.core.ButtonPrototype;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ActionComponent;
import com.ui.core.component.misc.ActionInProgressComponent;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.misc.TooltipComponent;
import com.ui.hud.shared.command.ResourceComponent;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.information.ResourceModalView;
import com.ui.modal.information.StatInformationView;
import com.ui.modal.shipyard.ComponentSelection;
import com.ui.modal.store.StoreView;
import com.util.CommonFunctionUtil;
import com.util.statcalc.StatCalcUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.events.MouseEvent;
import flash.events.TextEvent;
import flash.events.TimerEvent;
import flash.geom.Rectangle;
import flash.text.StyleSheet;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.Timer;
import flash.utils.getDefinitionByName;

import org.adobe.utils.StringUtil;
import org.shared.ObjectPool;

public class RefitBuildingView extends View
{
private var _bg:Bitmap;
private var _closeBtn:BitmapButton;
private var _moduleKey:Bitmap;
private var _statBg:ScaleBitmap;
private var _statHolder:Array;
private
```

```
var _refitEntityImage:ImageComponent;
private var _entityComponents:Dictionary; // String (slot name), ComponentSelection
private var _viewTitle:Label;
private var _statsTitle:Label;
private var _equippedTitle:Label;

private var _buildingVO:BuildingVO;
private var _modules:Dictionary;
private var _newComponents:Boolean = false;
private var _isTurret:Boolean = false;

private var _transaction:TransactionVO;

private var _buildComponent:ActionComponent;
private var _buildCostComponent:ResourceComponent;
private var _refitInProgressComponent:ActionInProgressComponent;
private var _toolTipComponent:TooltipComponent;
private var _currentRequirements:RequirementVO;

private var _statWindowString:String;

private var _infoBtn:BitmapButton;

private var _armorBar:ProgressBar;
private var _buildingDpsBar:ProgressBar;
private var _forceShieldingBar:ProgressBar;
private var _explosiveShieldingBar:ProgressBar;
private var _energyShieldingBar:ProgressBar;
private var _healthBar:ProgressBar;
private var _maskingBar:ProgressBar;
private var _profileBar:ProgressBar;

private var _buildingDpsLabel:Label;
private var _buildingDpsValue:Label;
private var _forceShieldingLabel:Label;
private var _forceShieldingValue:Label;
private var _explosiveShieldingLabel:Label;
private var _explosiveShieldingValue:Label;
private var _energyShieldingLabel:Label;
private var _energyShieldingValue:Label;
private var _healthLabel:Label;
private var _healthValue:Label;
private var _profileLabel:Label;
private var _profileValue:Label;
private var _armorLabel:Label;
private var _armorValue:Label;
private var _maskingLabel:Label;
private var _maskingValue:Label;

private var _equippedMod:IPrototype;

private
```

```
var _refitTimer:Timer;

private var _cost:String = 'CodeString.Shared.Cost' //Cost
private var _buildNow:String = 'CodeString.Shared.BuildNowBtn'; // Build Now
private var _build:String = 'CodeString.Shared.BuildBtn'; // Build
private var _refitNow:String = 'CodeString.Shared.RefitNowBtn'; // Refit Now
private var _refit:String = 'CodeString.Shared.RefitBtn'; // Refit
private var _stats:String = 'CodeString.Shared.Stats'; // Stats
private var _offline:String = 'CodeString.Docks.OfflineBtn'; // Offline
private var _cancelText:String = 'CodeString.Shared.CancelBtn'; //Cancel
private var _speedUpText:String = 'CodeString.ContextMenu.Starbase.SpeedUp'; //Speed Up

private var _unequipBtn:String = 'CodeString.BuildRefit.UnequipBtn' //Unequip
private var _unequip:String = 'CodeString.BuildRefit.Unequip' //Click the unequip button below to
remove the [[String.ModuleName]].
private var _module:String = 'CodeString.BuildRefit.Module'; // Module
private var _titleTurret:String = 'CodeString.BuildRefit.Title.Turret' //TURRET
private var _titleShield:String = 'CodeString.BuildRefit.Title.Shield'; //SHIELD

private var _getResourcesBtnText:String = 'CodeString.Shared.GetResources'; //GET
RESOURCES
private var _alertBodyBuyResources:String = 'CodeString.Alert.BuyResources.Body';
private var _alertHeadingBuyResources:String = 'CodeString.Alert.BuyResources.Title';
private var _emptySlot:String = 'CodeString.Shipyard.EmptySlot'; //Empty

private var _defenseProjectTitle:String = 'CodeString.Alert.BuildRefit.Title'; //Defense Project In
Progress
private var _defenseProjectAlertBody:String = 'CodeString.Alert.BuildRefit.Body'; //You have a
defense project currently in progress. Would you like to speed it up?
protected var _speedUpBtnText:String = 'CodeString.Shared.SpeedUp';
protected var _cancelBtnText:String = 'CodeString.Shared.CancelBtn';

[PostConstruct]
override public function init():void
{
super.init();

_bg = PanelFactory.getPanel("RefitWindowBGBMD");
_closeBtn = ButtonFactory.getCloseButton(_bg.width - 40, 25);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

var scaleRect:Rectangle = new Rectangle(0, 5, 299, 16);
_statBg = PanelFactory.getScaleBitmapPanel('RefitWindowStatsBGBMD', 299, 37, scaleRect);
_statBg.x = 350;
_statBg.y = 90;

_statHolder = new Array;

_refitEntityImage = new ImageComponent();
_refitEntityImage.init(128, 128);

_viewTitle
```

```
= new Label(28, 0xffffff, 200, 30);
_viewTitle.align = TextFormatAlign.LEFT;
_viewTitle.x = 35;
_viewTitle.y = 20;

var keyClass:Class;
switch (_buildingVO.asset)
{
case TypeEnum.SHIELD_GENERATOR:
_viewTitle.text = _titleShield;
_isTurret = false;
keyClass = Class(getDefinitionByName('DefenseKeyBMD'));
break;
case TypeEnum.POINT_DEFENSE_PLATFORM:
_viewTitle.text = _titleTurret;
_isTurret = true;
keyClass = Class(getDefinitionByName('WeaponKeyBMD'));
break;
}

presenter.loadIcon(AssetModel.instance.getEntityData(_buildingVO.asset).iconImage,
onImageLoaded);

_moduleKey = new Bitmap(BitmapData(new keyClass));
_moduleKey.x = 61;

if (_isTurret)
_moduleKey.y = 370;
else
_moduleKey.y = 375;


_statsTitle = new Label(20, 0xf0f0f0, 200, 30);
_statsTitle.align = TextFormatAlign.LEFT;
_statsTitle.constrictTextToSize = false;
_statsTitle.x = 350;
_statsTitle.y = 66;
_statsTitle.letterSpacing = 2;
_statsTitle.text = _stats;

var style:StyleSheet = new StyleSheet();
var hover:Object = new Object();
hover.color = "#b3ddf2";
var link:Object = new Object();
link.color = "#e7e7e7";

style.setStyle("a:link", link);
style.setStyle("a:hover", hover);

_equippedTitle = new Label(14, 0xe7e7e7, 240, 30);
_equippedTitle.align
```

```
= TextFormatAlign.LEFT;
_equippedTitle.constrictTextToSize = false;
_equippedTitle.x = 31;
_equippedTitle.y = 386;
_equippedTitle.letterSpacing = 1.5;
_equippedTitle.htmlText = _emptySlot;
addListener(_equippedTitle, TextEvent.LINK, linkEvent);
addListener(_equippedTitle, MouseEvent.ROLL_OVER, onTextRollOver);
addListener(_equippedTitle, MouseEvent.ROLL_OUT, onTextRollOut);
_equippedTitle.mouseEnabled = true;
_equippedTitle.styleSheet = style;

_buildComponent = new ActionComponent(new ButtonPrototype(_refit, onBuildClick), new
ButtonPrototype(_refitNow, onBuildNowClick), new ButtonPrototype(_getResourcesBtnText,
onClickCannotAffordResourceDialog),
new ButtonPrototype(_buildNow, popPaywall));
_buildComponent.x = 368;
_buildComponent.y = 376;

_buildCostComponent = ObjectPool.get(ResourceComponent);
_buildCostComponent.init(true, false, 35);
_buildCostComponent.x = 335;
_buildCostComponent.y = 301;

_refitInProgressComponent = new ActionInProgressComponent(new
ButtonPrototype(_cancelText, onCancelClick), new ButtonPrototype(_speedUpText,
openStoreToTransaction));
_refitInProgressComponent.visible = false;
_refitInProgressComponent.x = 349;
_refitInProgressComponent.y = 366;

_toolTipComponent = ObjectPool.get(TooltipComponent);
_toolTipComponent.init(2, 332, 278);

_infoBtn = ButtonFactory.getBitmapButton('TradeRouteInfoBtnNeutralBMD', 286, 69, '',
0xFFFFFF, 'TradeRouteInfoBtnRollOverBMD', 'TradeRouteInfoBtnDownBMD');
_infoBtn.addEventListener(MouseEvent.CLICK, showFullTooltip);


addChild(_bg);
addChild(_statBg);
addChild(_buildComponent);
addChild(_buildCostComponent);
addChild(_refitInProgressComponent);
addChild(_closeBtn);
addChild(_refitEntityImage);
addChild(_viewTitle);
addChild(_statsTitle);
addChild(_equippedTitle);
addChild(_infoBtn);
addChild(_moduleKey);
```

```
layoutComponentSelectors();

if (_isTurret)
{
createProgressBar("_buildingDpsBar", "_buildingDpsLabel", "_buildingDpsValue", 'shipDps',
366, 102);
createProgressBar("_healthBar", "_healthLabel", "_healthValue", 'health', 366, 129);
createProgressBar("_profileBar", "_profileLabel", "_profileValue", 'profile', 366, 156);
createProgressBar("_armorBar", "_armorLabel", "_armorValue", 'armor', 366, 183);
createProgressBar("_maskingBar", "_maskingLabel", "_maskingValue", 'masking', 366, 210);
_statBg.height += 110;
} else
{
createProgressBar("_forceShieldingBar", "_forceShieldingLabel", "_forceShieldingValue",
'forceShielding', 366, 102);
createProgressBar("_explosiveShieldingBar", "_explosiveShieldingLabel",
"_explosiveShieldingValue", 'explosiveShielding', 366, 129);
createProgressBar("_energyShieldingBar", "_energyShieldingLabel", "_energyShieldingValue",
'energyShielding', 366, 156);
createProgressBar("_healthBar", "_healthLabel", "_healthValue", 'health', 366, 183);
createProgressBar("_profileBar", "_profileLabel", "_profileValue", 'profile', 366, 210);
createProgressBar("_armorBar", "_armorLabel", "_armorValue", 'armor', 366, 237);
createProgressBar("_maskingBar", "_maskingLabel", "_maskingValue", 'masking', 366, 264);
_statBg.height += 164;
}


_refitTimer = new Timer(1000);
addListener(_refitTimer, TimerEvent.TIMER, updateTimer);

_transaction = presenter.getStarbaseBuildingTransaction(null, _buildingVO.id);

_modules = new Dictionary();
var slots:Array = _buildingVO.prototype.getValue('slots');
var i:int = 0;
if (_transaction)
{
if (_transaction.type == TransactionEvent.STARBASE_REFIT_BUILDING)
{
for (i = 0; i < slots.length; i++)
{
if (_buildingVO.refitModules.hasOwnProperty(slots[i]))
_modules[slots[i]] = _buildingVO.refitModules[slots[i]];

}

_refitInProgressComponent.visible = true;
_buildComponent.visible = false;
_buildCostComponent.visible
```

```
= false;

_refitTimer.start();
}
} else
{
//equip the current modules if any
for (i = 0; i < slots.length; i++)
{
if (_buildingVO.modules.hasOwnProperty(slots[i]))
_modules[slots[i]] = _buildingVO.modules[slots[i]];

}
}

//start off with the currently equipped modules
showEquipped();

addEffects();
effectsIN();
}

private function createProgressBar( barRef:String, labelRef:String, valueRef:String,
protoName:String, x:int, y:int ):void
{
var statHolderClass:Class = Class(getDefinitionByName('RefitWindowStatsBarBGBMD'));
var statHolder:Bitmap = new Bitmap(BitmapData(new statHolderClass));
statHolder.x = x - 9;
statHolder.y = y - 4;
addChild(statHolder);
_statHolder.push(statHolder);

var statProto:IPrototype = presenter.getStatPrototypeByName(protoName);
var defaultMaxValue:Number =
presenter.getConstantPrototypeValueByName("interfaceCalibrationDefaultStatValue");
var maxValue:Number = statProto.getUnsafeValue("stdMax");
maxValue = (maxValue) ? maxValue : defaultMaxValue;
this[barRef] = new ProgressBar();
this[barRef].init(ProgressBar.HORIZONTAL, new Bitmap(new BitmapData(248, 15, true,
0x994f82b4)), null, 0.01);
this[barRef].setMinMax(0, maxValue);
this[barRef].x = x;
this[barRef].y = y;
addChild(this[barRef]);

this[labelRef] = new Label(12, 0xFFFFFF, 248, 25, true, 1);
this[labelRef].align = TextFormatAlign.LEFT;
this[labelRef].text = statProto.getValue("lableLocKey");
this[labelRef].y = this[barRef].y - 3;
this[labelRef].x = this[barRef].x;
this[labelRef].constrictTextToSize
```

```
= false;
addChild(this[labelRef]);

this[valueRef] = new Label(12, 0xFFFFFF, 248, 25, true, 1);
this[valueRef].align = TextFormatAlign.RIGHT;
this[valueRef].y = this[barRef].y - 3;
this[valueRef].x = this[barRef].x;
this[valueRef].constrictTextToSize = false;
addChild(this[valueRef]);
}

private function setBarValue( stat:String, useStatCalc:Boolean, rating:Boolean ):void
{
// Get the stat value by calc or directly as needed
var statValue:Number;
if (useStatCalc && _buildingVO)
statValue = StatCalcUtil.entityStatCalc(_buildingVO, stat);
else
statValue = _buildingVO[stat];


// Format the value and turn it into a string
var statProto:IPrototype = presenter.getStatPrototypeByName(stat);
var statString:String = StringUtil.formatValue(String(statValue), statProto, "flat", "base");
this["_" + stat + "Value"].setTextWithTokens(statProto.getValue("valueLocKey"),
{'[[Value]]':statString});

// Show modifier if this was a rating
if (rating)
{
var loc:Localization = Localization.instance;
var modProto:IPrototype = presenter.getStatPrototypeByName("genericPercent");
var percentString:String = modProto.getValue("valueLocKey");
var modValue:Number = CommonFunctionUtil.ratingToModifier(statValue);
var modString:String = StringUtil.formatValue(String(modValue), modProto, "flat", "base");
this["_" + stat + "Value"].text += loc.getStringWithTokens(percentString, {'[[Value]]':modString});
}

// Set the bar value
this["_" + stat + "Bar"].amount = statValue;
}

private function onCancelClick( e:MouseEvent ):void
{
if (_transaction)
{
//cancel the refit
presenter.cancelTransaction(_transaction);
destroy();
}
}
```

```
private function onBuildClick( e:MouseEvent ):void
{
//do refitting
if (_newComponents)
{
var transaction:TransactionVO = presenter.getStarbaseBuildingTransaction('Defense');
if (transaction)
{
popBusyDialog();
} else
{
presenter.performTransaction(TransactionEvent.STARBASE_REFIT_BUILDING, _buildingVO,
PurchaseTypeEnum.NORMAL, _modules);
destroy();
}
}
}

private function onBuildNowClick( e:MouseEvent ):void
{
if (_newComponents)
{
var transaction:TransactionVO = presenter.getStarbaseBuildingTransaction('Defense');
if (transaction)
{
popBusyDialog();
} else
{
presenter.performTransaction(TransactionEvent.STARBASE_REFIT_BUILDING, _buildingVO,
PurchaseTypeEnum.INSTANT, _modules);
destroy();
}
}
}

private function onRefitWithResourcePurchase():void
{
if (_newComponents)
{
var transaction:TransactionVO = presenter.getStarbaseBuildingTransaction('Defense');
if (transaction)
{
popBusyDialog();
} else
{
presenter.performTransaction(TransactionEvent.STARBASE_REFIT_BUILDING, _buildingVO,
PurchaseTypeEnum.GET_RESOURCES, _modules);
destroy();
}
```

```
}
}

private function popBusyDialog():void
{
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_speedUpBtnText, speedUpDefenseTransaction, null, true,
ButtonEnum.GOLD_A));
buttons.push(new ButtonPrototype(_cancelBtnText));
showConfirmation(_speedUpBtnText, _defenseProjectAlertBody, buttons);
}

private function speedUpDefenseTransaction():void
{
var transaction:TransactionVO = presenter.getStarbaseBuildingTransaction('Defense');
if (transaction)
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
nStoreView.setSelectedTransaction(transaction);
}
}

private function updateTimer( e:TimerEvent ):void
{
if (_transaction)
{
var buildTime:int = _transaction.timeRemainingMS;
_refitInProgressComponent.timeRemaining = buildTime;
if (buildTime <= 0)
clearTimer();
} else
clearTimer();

}

private function clearTimer():void
{
_refitTimer.stop();
}

private function openStoreToTransaction( e:MouseEvent ):void
{
if (_transaction)
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
nStoreView.setSelectedTransaction(_transaction);
destroy();
}
```

```
}

private function onPurchaseMoreResources( args:Array ):void
{
if (args && args.length > 0)
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
nStoreView.openToResourcesAndFilter(args[0]);
}
}

private function onSelectComponent( slotId:String, index:uint,
currentComponentSelection:ComponentSelection ):void
{
var basicSlotType:String = presenter.getSlotType(currentComponentSelection.slotName);
var constructionView:ConstructionView =
ConstructionView(_viewFactory.createView(ConstructionView));
constructionView.openOn(ConstructionView.COMPONENT, basicSlotType, null);
_viewFactory.notify(constructionView);
}

protected function onClickCannotAffordResourceDialog( e:MouseEvent ):void
{
if (_currentRequirements.purchaseVO.canPurchaseResourcesWithPremium)
{
var purchaseVO:PurchaseVO = _currentRequirements.purchaseVO;
var view:ResourceModalView =
ResourceModalView(_viewFactory.createView(ResourceModalView));
_viewFactory.notify(view);
view.setUp(purchaseVO.creditsAmountShort, purchaseVO.alloyAmountShort,
purchaseVO.energyAmountShort, purchaseVO.syntheticAmountShort,
'CodeString.Alert.BuyResources.Title', 'CodeString.Alert.BuyResources.Body',
false, onRefitWithResourcePurchase, purchaseVO.resourcePremiumCost);
} else
popPaywall();
}

public function onModuleSelected( component:IPrototype ):void
{
var slot:String = _buildingVO.getValue("slots")[0];
_buildingVO.equipRefitModule(component, slot);
_modules[slot] = component;
showEquipped();
}

private function showEquipped():void
{
var slots:Array = _buildingVO.prototype.getValue('slots');
var
```

```
assetVO:AssetVO;
var kalganCost:int = 0;
var title:String = '';
var description:String = '';
var details:String = '';
var proto:IPrototype;
_newComponents = false;

_currentRequirements = null;
//note the UI was only designed to work with 1 module
for (var slot:String in _modules)
{
if (_modules.hasOwnProperty(slot))
{
proto = _modules[slot];
if (proto)
{
if (!_buildingVO.modules.hasOwnProperty(slot) || proto != _buildingVO.modules[slot])
_newComponents = true;

assetVO = presenter.getAssetVO(proto);
title = assetVO.visibleName;

// _description.text = assetVO.descriptionText;
} else
{
// Player has cleared the slot. This only works because there is only one slot.
// _description.text = "";
_buildCostComponent.visible = false;
if (slot in _buildingVO.modules && proto != _buildingVO.modules[slot])
_newComponents = true;
}

(_entityComponents[slot] as ComponentSelection).selectedComponent = proto;
}
_currentRequirements =
presenter.getRequirements(TransactionEvent.STARBASE_REFIT_BUILDING, _buildingVO);
}


if (proto)
{
if (_newComponents)
{
_buildComponent.timeCost = (_buildingVO.buildTimeSeconds > 0) ?
_buildingVO.buildTimeSeconds : 0;

if (_currentRequirements)
{
_buildCostComponent.updateCost(_buildingVO.alloyCost,
(_currentRequirements.purchaseVO.alloyAmountShort
```

```
== 0), CurrencyEnum.ALLOY);
_buildCostComponent.updateCost(_buildingVO.creditsCost,
(_currentRequirements.purchaseVO.creditsAmountShort == 0), CurrencyEnum.CREDIT);
_buildCostComponent.updateCost(_buildingVO.energyCost,
(_currentRequirements.purchaseVO.energyAmountShort == 0), CurrencyEnum.ENERGY);
_buildCostComponent.updateCost(_buildingVO.syntheticCost,
(_currentRequirements.purchaseVO.syntheticAmountShort == 0), CurrencyEnum.SYNTHETIC);

_buildComponent.requirements = _currentRequirements;
}

_buildComponent.instantCost = _currentRequirements.purchaseVO.premium;

if (!_transaction)
_buildCostComponent.visible = true;
_buildComponent.actionBtnText = _build;
_buildComponent.instantActionBtnText = _buildNow;

} else
_buildCostComponent.visible = false;

_statWindowString = StringUtil.getTooltip(proto.getValue('type'), proto, false);

_equippedMod = proto;

} else
{
_buildComponent.timeCost = 0;
_buildCostComponent.visible = false;
_infoBtn.visible = false;
}

if (_isTurret)
setBarValue("buildingDps", false, false);
else
{
setBarValue("forceShielding", false, false);
setBarValue("explosiveShielding", false, false);
setBarValue("energyShielding", false, false);
}
setBarValue("health", true, false);
setBarValue("profile", true, true);
setBarValue("armor", true, true);
setBarValue("masking", true, true);

_buildComponent.enabled = _newComponents;

title = Localization.instance.getString(title);
if (title == '')
{
title
```

```
= Localization.instance.getString(_emptySlot);
_equippedTitle.htmlText = '<a href="event:' + title + '">' + title.toUpperCase() + '</a>';
} else
{
_equippedTitle.htmlText = '<a href="event:' + title + '">' + title.toUpperCase() + '</a>';
_equippedTitle.textColor = _entityComponents[slots[0]].rarityColor;
}



}

private function linkEvent( event:TextEvent ):void
{
var slots:Array = _buildingVO.prototype.getValue('slots');
var cs:ComponentSelection = _entityComponents[slots[0]];

onSelectComponent(cs.slotType, 0, cs);
}

private function layoutComponentSelectors():void
{
_entityComponents = new Dictionary;
var slots:Array = _buildingVO.prototype.getValue('slots');
var spaceX:int = 50;
var spaceY:int = 50;

var type:String;
var cs:ComponentSelection;
var xpos:Number = 109;
var ypos:Number = 168;

if (slots[0].indexOf(SlotComponentEnum.SLOT_TYPE_TECH) != -1)
type = SlotComponentEnum.SLOT_TYPE_TECH;
else if (slots[0].indexOf(SlotComponentEnum.SLOT_TYPE_DEFENSE) != -1)
type = SlotComponentEnum.SLOT_TYPE_DEFENSE;
else if (slots[0].indexOf(SlotComponentEnum.SLOT_TYPE_SPECIAL) != -1)
type = SlotComponentEnum.SLOT_TYPE_SPECIAL;
else if (slots[0].indexOf(SlotComponentEnum.SLOT_TYPE_WEAPON) != -1)
type = SlotComponentEnum.SLOT_TYPE_WEAPON;
else if (slots[0].indexOf(SlotComponentEnum.SLOT_TYPE_SHIELD) != -1)
type = SlotComponentEnum.SLOT_TYPE_DEFENSE;
cs = new ComponentSelection(type, slots[0], 0);
cs.x = 171;
cs.y = 230;
xpos += cs.width;
cs.onSelectComponent.add(onSelectComponent);
cs.onHover.add(highlightPrototypeLabel);
_entityComponents[slots[0]] = cs;
addChild(cs);
}

private
```

```
function onTextRollOver( e:MouseEvent ):void
{
if (_buildingVO)
{
var slots:Array = _buildingVO.prototype.getValue('slots');
if (slots && slots.length > 0)
{
Label(e.target).textColor = 0xb3ddf2;
_entityComponents[slots[0]].onOutsideRollOver();
}
}

}

private function onTextRollOut( e:MouseEvent ):void
{
if (_buildingVO)
{
var slots:Array = _buildingVO.prototype.getValue('slots');
if (slots && slots.length > 0)
{
Label(e.target).textColor = _entityComponents[slots[0]].rarityColor;
_entityComponents[slots[0]].onOutsideRollOut();
}
}
}

private function highlightPrototypeLabel( rollover:Boolean, index:uint ):void
{
var slots:Array = _buildingVO.prototype.getValue('slots');
var color:uint;
if (rollover)
color = 0xb3ddf2;
else
color = _entityComponents[slots[0]].rarityColor;

_equippedTitle.textColor = color;
}

private function showFullTooltip( e:MouseEvent ):void
{
_toolTipComponent.layoutTooltip(_statWindowString, 25, 68, 15, 6);

var view:StatInformationView =
StatInformationView(_viewFactory.createView(StatInformationView));
view.SetUp(_toolTipComponent);
_viewFactory.notify(view);
}

private function onImageLoaded( asset:BitmapData ):void
{
```

```
if (_refitEntityImage)
{
_refitEntityImage.onImageLoaded(asset);
_refitEntityImage.x = 171 - (_refitEntityImage.width * 0.5);
_refitEntityImage.y = 230 - (_refitEntityImage.height * 0.5);
}
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

public function set buildingVO( vo:BuildingVO ):void
{
_buildingVO = vo;
_buildingVO.calculateCosts();
}

private function popPaywall( e:MouseEvent = null ):void
{
CommonFunctionUtil.popPaywall();
}

[Inject]
public function set presenter( value:IStarbasePresenter ):void { _presenter = value; }
public function get presenter():IStarbasePresenter { return IStarbasePresenter(_presenter); }

override public function get typeUnique():Boolean { return false; }

override public function destroy():void
{
super.destroy();

_bg = null;
_closeBtn.destroy();
_closeBtn = null;
_moduleKey = null;
_statBg = null;

_refitEntityImage.destroy();
_refitEntityImage = null;

for each (var cs:ComponentSelection in _entityComponents)
{
cs.onSelectComponent.remove(onSelectComponent);
cs.destroy();
}
_entityComponents = null;

_viewTitle.destroy();
_viewTitle
```

```
= null;

_statsTitle.destroy();
_statsTitle = null;

_equippedTitle.destroy();
_equippedTitle = null;

_buildingVO = null;
_modules = null;

_buildComponent.destroy();
_buildComponent = null;

ObjectPool.give(_buildCostComponent);
_buildCostComponent = null;

_refitInProgressComponent.destroy();
_refitInProgressComponent = null;

ObjectPool.give(_toolTipComponent);
_toolTipComponent = null;

_currentRequirements = null;

_statWindowString = null;

_infoBtn.destroy();
_infoBtn = null;

_armorBar.destroy();
_armorBar = null;

if (_isTurret)
{
_buildingDpsBar.destroy();
_buildingDpsBar = null;

_buildingDpsLabel.destroy();
_buildingDpsLabel = null;

_buildingDpsValue.destroy();
_buildingDpsValue = null;
} else
{
_forceShieldingBar.destroy();
_forceShieldingBar = null;

_explosiveShieldingBar.destroy();
_explosiveShieldingBar = null;

_energyShieldingBar.destroy();
```

```
_energyShieldingBar = null;

_forceShieldingLabel.destroy();
_forceShieldingLabel = null;

_forceShieldingValue.destroy();
_forceShieldingValue = null;

_explosiveShieldingLabel.destroy();
_explosiveShieldingLabel = null;

_explosiveShieldingValue.destroy();
_explosiveShieldingValue = null;

_energyShieldingLabel.destroy();
_energyShieldingLabel = null;

_energyShieldingValue.destroy();
_energyShieldingValue = null;
}

_healthBar.destroy();
_healthBar = null;

_maskingBar.destroy();
_maskingBar = null;

_profileBar.destroy();
_profileBar = null;

_healthLabel.destroy();
_healthLabel = null;

_healthValue.destroy();
_healthValue = null;

_profileLabel.destroy();
_profileLabel = null;

_profileValue.destroy();
_profileValue = null;

_armorLabel.destroy();
_armorLabel = null;

_armorValue.destroy();
_armorValue = null;

_maskingLabel.destroy();
_maskingValue
```

```
= null;

_equippedMod = null;
}
}
}

------------------------------------
File 797: igw\com\ui\modal\building\RepairBaseView.as
package com.ui.modal.building
{
import com.enum.server.PurchaseTypeEnum;
import com.event.TransactionEvent;
import com.model.asset.AssetModel;
import com.presenter.starbase.IStarbasePresenter;
import com.service.language.Localization;
import com.ui.core.ButtonPrototype;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ActionComponent;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.misc.TooltipComponent;
import com.ui.modal.ButtonFactory;
import com.util.CommonFunctionUtil;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.shared.ObjectPool;

public class RepairBaseView extends View
{
public static var PLAYER_KNOWS_ABOUT_REPAIR:Boolean = false;

protected var _actionComponent:ActionComponent;
protected var _bg:Sprite;
protected var _buildingInfo:Label;
protected var _closeBtn:BitmapButton;
protected var _descriptionInfo:Label;
protected var _image:ImageComponent;
protected var _tooltipComponent:TooltipComponent;
protected var _viewName:Label;

private var _repair:String = "CodeString.Docks.RepairBtn"; //REPAIR
private var _repairNow:String = "CodeString.Docks.RepairNowBtn"; //REPAIR NOW
private var _getResources:String = 'CodeString.Shared.GetResources'; //GET RESOURCES
private var _getPalladium:String = 'CodeString.Shared.GetPalladium' //GET PALLADIUM

private
```

```
var _damageReportText:String = "CodeString.RepairBaseView.DamageReport"; //Damage
Report
private var _damageReportDescription:String =
"CodeString.RepairBaseView.DamageReportDescription"; //Your base has been attacked and
needs repairs!
private var _damageTooltip:String = 'CodeString.RepairBaseView.Tooltip.Damage';
//<textformat blockindent="6" rightmargin="13"><font size="13" color="#B3DDF2">Damage
Taken:</font><font size="13" color="#f0f0f0">
[[Number.TotalBaseDamage]]%</font><br/>\n<font size="13" color="#B3DDF2">Damaged
Buildings:</font><font size="13" color="#f0f0f0">
[[Number.TotalDamagedBuildings]]</font><br/>\n<font size="13" color="#B3DDF2">Destroyed
Buildings:</font><font size="13" color="#f0f0f0">
[[Number.TotalDestroyedBuildings]]</font><br/>\n</textformat>

[PostConstruct]
override public function init():void
{
super.init();
var windowFrameBGClass:Class =
Class(getDefinitionByName("BuildWindowMaxLevelWindowNoSalvageBGMC"));
_bg = Sprite(new windowFrameBGClass());
addChild(_bg);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);
addChild(_closeBtn);

_image = new ImageComponent();
_image.init(150, 150);
_image.x = 51;
_image.y = 108;
addChild(_image);

_viewName = new Label(30, 0xFFFFFF, _bg.width - 100, 40);
_viewName.y = 17;
_viewName.x = 29;
_viewName.align = TextFormatAlign.LEFT;
_viewName.text = _repair;
addChild(_viewName);

_buildingInfo = new Label(20, 0xFFFFFFF, 434, 84);
_buildingInfo.x = 35;
_buildingInfo.y = 65;
_buildingInfo.multiline = false;
_buildingInfo.align = TextFormatAlign.LEFT;
_buildingInfo.text = _damageReportText;
addChild(_buildingInfo);

_descriptionInfo = new Label(13, 0xFFFFFF, 434, 84, true, 1);
_descriptionInfo.x = 200;
_descriptionInfo.y
```

```
= 115;
_descriptionInfo.multiline = true;
_descriptionInfo.align = TextFormatAlign.LEFT;
_descriptionInfo.text = _damageReportDescription;
addChild(_descriptionInfo);

_actionComponent = new ActionComponent(new ButtonPrototype(_repair, onActionBtnClick),
new ButtonPrototype(_repairNow, onActionInstantBtnClick),
new ButtonPrototype(_getResources, onActionBtnClick),
new ButtonPrototype(_getPalladium, popPaywall));
_actionComponent.x = 320;
_actionComponent.y = 280;
_actionComponent.visible = true;
_actionComponent.timeCost = presenter.getRepairTime();
_actionComponent.instantCost = presenter.getRepairCost();
_actionComponent.instantActionBtnEnabled = false;
_actionComponent.instantActionBtn.visible = false;
addChild(_actionComponent);

//stats
// Add indentation
var tooltip:String = Localization.instance.getStringWithTokens(_damageTooltip,
{"[[Number.TotalBaseDamage]]":presenter.totalBaseDamage,
"[[Number.TotalDamagedBuildings]]":presenter.
totalDamagedBuildings,
"[[Number.TotalDestroyedBuildings]]":presenter.totalDestroyedBuildings});
tooltip = tooltip.split('<br/>').join('<br/>\n');
_tooltipComponent = ObjectPool.get(TooltipComponent);
_tooltipComponent.init(1, 300);
_tooltipComponent.layoutTooltip(tooltip, 50, 277);
addChild(_tooltipComponent);

PLAYER_KNOWS_ABOUT_REPAIR = true;

AssetModel.instance.getFromCache("assets/CommandCenter5_Icon.png",
_image.onImageLoaded);

addEffects();
effectsIN();
removeChild(_closeBtn);
}

protected function onActionBtnClick( e:MouseEvent ):void
{
PLAYER_KNOWS_ABOUT_REPAIR = false;
presenter.performTransaction(TransactionEvent.STARBASE_REPAIR_BASE, null,
PurchaseTypeEnum.NORMAL);
destroy();
}
protected function onActionInstantBtnClick( e:MouseEvent ):void
{
```

```
PLAYER_KNOWS_ABOUT_REPAIR = false;
presenter.performTransaction(TransactionEvent.STARBASE_REPAIR_BASE, null,
PurchaseTypeEnum.INSTANT);
destroy();
}

private function popPaywall( e:MouseEvent = null ):void
{
CommonFunctionUtil.popPaywall();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IStarbasePresenter ):void { _presenter = value; }
public function get presenter():IStarbasePresenter { return IStarbasePresenter(_presenter); }

override public function get typeUnique():Boolean { return false; }

override public function destroy():void
{
if(!PLAYER_KNOWS_ABOUT_REPAIR)
{
super.destroy();

_bg = null;
_closeBtn.destroy();
_closeBtn = null;

_image.destroy();
_image = null;

if (_buildingInfo)
{
_buildingInfo.destroy();
_buildingInfo = null;
}

if (_descriptionInfo)
{
_descriptionInfo.destroy();
_descriptionInfo = null;
}

if (_viewName)
{
_viewName.destroy();
_viewName = null;
}
```

```
ObjectPool.give(_tooltipComponent);
_tooltipComponent = null;
}
}
}
}
```

File 798: igw\com\ui\modal\construction\ConstructionInfoView.as

```
package com.ui.modal.construction
{
import com.enum.AudioEnum;
import com.controller.transaction.requirements.CategoryNotBuildingRequirement;
import com.controller.transaction.requirements.IRequirement;
import com.controller.transaction.requirements.PurchaseVO;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.CurrencyEnum;
import com.enum.StarbaseConstructionEnum;
import com.enum.TypeEnum;
import com.enum.server.PurchaseTypeEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.TransactionEvent;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.model.starbase.BuildingVO;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.IConstructionPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ButtonPrototype;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;
import com.ui.core.component.label.RequirementLabel;
import com.ui.core.component.misc.ActionComponent;
import com.ui.core.component.misc.BlueprintActionComponent;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.misc.TooltipComponent;
import com.ui.hud.shared.command.ResourceComponent;
import com.ui.modal.ButtonFactory;
import com.ui.modal.information.ResourceModalView;
import com.ui.modal.information.StatInformationView;
import
```

```
com.ui.modal.store.StoreView;
import com.util.CommonFunctionUtil;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TextEvent;
import flash.filters.GlowFilter;
import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;
import org.shared.ObjectPool;

public class ConstructionInfoView extends View
{
private var _actionComponent:ActionComponent;
private var _bg:DefaultWindowBG;
private var _blueprint:BlueprintVO;
private var _blueprintActionComponent:BlueprintActionComponent;
private var _blueprintActionComponent_complete:BlueprintActionComponent;
private var _callback:Function;
private var _closeButton:BitmapButton;
private var _description:Label;
private var _image:ImageComponent;
private var _imageFrame:ScaleBitmap;
private var _infoPanel:Sprite;
private var _infoBtn:BitmapButton;
private var _soundBtn:BitmapButton;
private var _prototype:IPrototype;
private var _requirements:RequirementVO;
private var _requirementsBG:ScaleBitmap;
private var _requirementLabels:Vector.<RequirementLabel>;
private var _requirementsPanel:Sprite;
private var _resourceComponent:ResourceComponent;
private var _specialButton:BitmapButton;
private var _state:int;
private var _statsPanel:Sprite;
private var _statWindowString:String;
private var _title:Label;
private var _tooltipComponent:TooltipComponent;
private var _fullStatTooltipComponent:TooltipComponent;

protected var _soundToPlay:String;

//code strings for localization
private var _buildNow:String = 'CodeString.Shared.BuildNowBtn'; // Build Now
private var _build:String = 'CodeString.Shared.BuildBtn'; //Build
private var _getResources:String = 'CodeString.Shared.GetResources'; //GET RESOURCES
private var _research:String = 'CodeString.Build.ResearchBtn';
private var _researchNow:String = 'CodeString.ResearchInformation.ResearchNowBtn';
//Research Now
private
```

```
var _upgrade:String = 'CodeString.BuildUpgrade.UpgradeBtn'; // Upgrade
private var _upgradeNow:String = 'CodeString.BuildUpgrade.UpgradeNowBtn'; // Upgrade Now
private var _buildProjectTitle:String = 'CodeString.Alert.BuildInProgress.Title'; //Build Project In Progress
private var _buildProjectAlertBody:String = 'CodeString.Alert.BuildInProgress.Body'; //You have a build project currently in progress. Would you like to speed it up?
private var _speedUpBtnText:String = 'CodeString.Shared.SpeedUp';
private var _cancelBtnText:String = 'CodeString.Shared.CancelBtn';
private var _contructionViewClose:String = 'CodeString.ConstructionView.Close'; //CLOSE
private var _contructionViewInfo:String = 'CodeString.ConstructionView.Info'; //INFO
private var _upgradeTitle:String = 'CodeString.BuildUpgrade.Title.Upgrade'; //UPGRADE
private var _detailsTitle:String = 'CodeString.ConstructionItem.Details'; //DETAILS
private var _buildTitle:String = 'CodeString.ConstructionItem.Build'; //BUILD
private var _researchTitle:String = 'CodeString.Research.Title'; //RESEARCH
private var _salvageTitle:String = 'CodeString.Shared.SalvageBtn'; //SALVAGE
private var _statsHeading:String = 'CodeString.Shipyard.StatsHeading'; //STATS
private var _requirementsHeading:String = 'CodeString.Shared.Requirements'; //REQUIREMENTS
private var _completeBtnText:String = 'CodeString.Shared.CompleteBtn'; //COMPLETE
private var _purchaseOneText:String = 'CodeString.Shared.PurchaseOneBtn'; //PURCHASE ONE
private var _completeResearchBtnText:String = 'CodeString.Shared.CompleteResearchBtn'; //COMPLETE RESEARCH

[Inject]
override public function init():void
{
super.init();
_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(688, 308);

_closeButton = UIFactory.getButton(ButtonEnum.BLUE_A, 240, 40, _bg.width - 263, _bg.height + 9, _contructionViewClose);

setupInfoPanel();
if (setupStatsPanel())
setupRequirementsPanel();

_closeButton.y = _bg.height + 9;

addChild(_bg);
addChild(_closeButton);
if (_specialButton)
{
_specialButton.y = _closeButton.y;
addChild(_specialButton);
}
addChild(_infoPanel);
addChild(_statsPanel);
addChild(_description);
addChild(_image);
```

```
addChild(_imageFrame);
addChild(_title);
addChild(_tooltipComponent);

if (_requirementsPanel)
{
addChild(_requirementsPanel);
addChild(_requirementsBG);
addChild(_resourceComponent);
addChild(_actionComponent);
if (_blueprintActionComponent)
addChild(_blueprintActionComponent);

if(_blueprintActionComponent_complete)
addChild(_blueprintActionComponent_complete);

for (var i:int = 0; i < _requirementLabels.length; i++)
{
if (_requirementLabels[i].lbl.htmlText != "")
{
if (_requirementLabels[i].showLink)
{
_requirementLabels[i].lbl.addEventListener(TextEvent.LINK, onClickLink);
_requirementLabels[i].lbl.mouseEnabled = true;
_requirementLabels[i].lbl.styleSheet = LabelFactory.linkStyleSheet;
}
addChild(_requirementLabels[i]);
}
}
}

addListener(_bg.closeButton, MouseEvent.CLICK, onClose);
addListener(_closeButton, MouseEvent.CLICK, onClose);
if (_specialButton)
addListener(_specialButton, MouseEvent.CLICK, onSpecialClicked);

_infoBtn = ButtonFactory.getBitmapButton('TradeRouteInfoBtnNeutralBMD', 111, 88, '',
0xFFFFFF, 'TradeRouteInfoBtnRollOverBMD', 'TradeRouteInfoBtnDownBMD');
_infoBtn.addEventListener(MouseEvent.CLICK, showFullTooltip);
addChild(_infoBtn);

if (_soundToPlay != null)
{
_soundBtn = ButtonFactory.getBitmapButton('TradeRouteInfoBtnNeutralBMD', 111, 166, '',
0xFFFFFF, 'TradeRouteInfoBtnRollOverBMD', 'TradeRouteInfoBtnDownBMD');
_soundBtn.addEventListener(MouseEvent.CLICK, playVOSound);
addChild(_soundBtn);
}

addEffects();
```

```
effectsIN();
}

public function setup( state:int, prototype:IPrototype ):void
{
_prototype = prototype;
_state = state;
}

private function setupInfoPanel():void
{
_infoPanel = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED,
PanelEnum.HEADER_NOTCHED, 667, 118, 32, _bg.bg.x + 15, _bg.bg.y + 5,
_contructionViewInfo);

_image = ObjectPool.get(ImageComponent);
_image.init(100, 100);
_image.center = true;
_image.x = _infoPanel.x + 9;
_image.y = _infoPanel.y + 41;

_imageFrame = UIFactory.getPanel(PanelEnum.CHARACTER_FRAME, 100, 100, _image.x,
_image.y);

var assetVO:AssetVO = presenter.getAssetVO(_prototype);

if(false && assetVO.key != null && assetVO.key.length > 0)
_soundToPlay = AudioEnum.VO_INFO_BASE_DIRECTORY + assetVO.key +
AudioEnum.VO_INFO_BASE_FORMAT;

presenter.loadImage(assetVO.mediumImage, _image.onImageLoaded);

_title = UIFactory.getLabel(LabelEnum.TITLE, 300, 45, _image.x + 108, _image.y - 2);
_title.align = TextFormatAlign.LEFT;
_title.useLocalization = false;
_title.bold = false;
_title.constrictTextToSize = true;
_title.text = Localization.instance.getString(assetVO.visibleName).toUpperCase();

_description = UIFactory.getLabel(LabelEnum.DESCRIPTION, 525, 75, _title.x, _title.y +
_title.height - 15);
_description.fontSize = 14;
_description.leading = -2;
_description.text = assetVO.descriptionText;
}

private function setupStatsPanel():Boolean
{
var proto:IPrototype;
var
```

```
showRequirements:Boolean = true;
_statsPanel = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED,
PanelEnum.HEADER_NOTCHED, 667, 110, 32, _infoPanel.x, _infoPanel.y + _infoPanel.height
+ 4, _statsHeading);

_tooltipComponent = ObjectPool.get(TooltipComponent);
_tooltipComponent.init(3, _statsPanel.width - 8, 92);
_tooltipComponent.x = _statsPanel.x + 4;
_tooltipComponent.y = _statsPanel.y + 36;
_tooltipComponent.mouseEnabled = _tooltipComponent.mouseChildren = false;

_fullStatTooltipComponent = ObjectPool.get(TooltipComponent);
_fullStatTooltipComponent.init(2, 332, 278);
_fullStatTooltipComponent.mouseEnabled = _fullStatTooltipComponent.mouseChildren = false;

switch (_state)
{
case ConstructionView.BUILD:
var count:int = presenter.getBuildingCount(_prototype.itemClass);
var maxCount:int = presenter.getBuildingMaxCount(_prototype.itemClass);
var building:BuildingVO = new BuildingVO();
proto = _prototype;
if (_prototype is BuildingVO)
{
if (presenter.getBuildingUpgrade(_prototype.getValue('upgrade')) != null)
{
_bg.addTitle(_upgradeTitle, 300);
proto = presenter.getBuildingUpgrade(_prototype.getValue('upgrade'));
building.prototype = proto;
_requirements =
presenter.getRequirements(TransactionEvent.STARBASE_BUILDING_UPGRADE, building);
} else
{
_bg.addTitle(_detailsTitle, 300);
showRequirements = false;
}
} else
{
if (count < maxCount)
{
_bg.addTitle(_buildTitle, 300);
building.prototype = proto;
_requirements = presenter.getRequirements(TransactionEvent.STARBASE_BUILDING_BUILD,
building);
} else if (_prototype.getValue('constructionCategory') != StarbaseConstructionEnum.PLATFORM
&&
presenter.getBuildingVOByClass(_prototype.itemClass, true) &&
presenter.getBuildingVOByClass(_prototype.itemClass, true).level != 10)
{
_bg.addTitle(_upgradeTitle, 300);
_prototype
```

```
= presenter.getBuildingVOByClass(_prototype.itemClass, true);
proto = presenter.getBuildingUpgrade(_prototype.getValue('upgrade'));
building.prototype = proto;
_requirements =
presenter.getRequirements(TransactionEvent.STARBASE_BUILDING_UPGRADE, building);
} else
{
_bg.addTitle(_detailsTitle, 300);
showRequirements = false;
}
}

_statWindowString = StringUtil.getTooltip(_prototype.getValue("type"), _prototype is BuildingVO
? BuildingVO(_prototype).prototype : _prototype, false);

_title.text += " " + count + "/" + maxCount;
_tooltipComponent.layoutTooltip(StringUtil.getTooltip(_prototype.getValue("type"), proto is
BuildingVO ? BuildingVO(proto).prototype : proto, false, (proto == _prototype) ? null :
_prototype));
if (_prototype is BuildingVO && _prototype.getValue("canBeRecycled") == true)
_specialButton = UIFactory.getButton(ButtonEnum.RED_A, 240, 40, _closeButton.x - 250,
_closeButton.y, _salvageTitle);
break;

case ConstructionView.RESEARCH:
var requirementMet:Boolean = presenter.requirementsMet(_prototype);
var isResearched:Boolean = presenter.isResearched(_prototype.name);
if (isResearched || !requirementMet)
{
if (isResearched)
showRequirements = false;
_bg.addTitle(_detailsTitle, 300);
} else if (requirementMet)
_bg.addTitle(_researchTitle, 300);

//see if this is a blueprint
var blueprint:BlueprintVO = presenter.getBlueprint(_prototype.name);
if (blueprint && (CONFIG::IS_CRYPTO || blueprint.partsCollected < blueprint.totalParts))
_title.text += " " + blueprint.partsCollected + "/" + blueprint.totalParts;
if (blueprint)
_requirements =
presenter.getRequirements(TransactionEvent.STARBASE_BLUEPRINT_PURCHASE,
_prototype);
else
_requirements = presenter.getRequirements(TransactionEvent.STARBASE_RESEARCH,
_prototype);

var rarity:String = _prototype.getUnsafeValue('rarity');
if (rarity != 'Common')
{
var
```

```
glow:GlowFilter = CommonFunctionUtil.getRarityGlow(rarity);
_title.textColor = glow.color;
_imageFrame.filters = [glow];
}

proto = presenter.getResearchItemPrototypeByName(_prototype.getValue("referenceName"));
_tooltipComponent.layoutTooltip(StringUtil.getTooltip(proto.getValue("type"), proto, false));
_statWindowString = StringUtil.getTooltip(proto.getValue("type"), proto, false);
break;
}

if (!showRequirements)
{
_statsPanel = UIFactory.destroyPanel(_statsPanel);
_statsPanel = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_DOUBLE_NOTCHED,
PanelEnum.HEADER_NOTCHED, 667, 110, 32, _infoPanel.x, _infoPanel.y + _infoPanel.height
+ 4, _statsHeading);
}

return showRequirements;
}

private function setupRequirementsPanel():void
{
_bg.setBGSize(688, 498);
_requirementLabels = new Vector.<RequirementLabel>;
_requirementsPanel =
UIFactory.getHeaderPanel(PanelEnum.CONTAINER_DOUBLE_NOTCHED,
PanelEnum.HEADER_NOTCHED, 667, 155, 32, _infoPanel.x, _statsPanel.y +
_statsPanel.height + 4, _requirementsHeading);

_requirementsBG = UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL,
646, 61, _requirementsPanel.x + 9, _requirementsPanel.y + 40);
_resourceComponent = ObjectPool.get(ResourceComponent);
_resourceComponent.init(true, false, 35);
_resourceComponent.x = _requirementsBG.x;
_resourceComponent.y = _requirementsBG.y + _requirementsBG.height + 9;

_resourceComponent.updateCost(_requirements.purchaseVO.alloyCost,
_requirements.purchaseVO.alloyAmountShort == 0, CurrencyEnum.ALLOY);
_resourceComponent.updateCost(_requirements.purchaseVO.creditsCost,
_requirements.purchaseVO.creditsAmountShort == 0, CurrencyEnum.CREDIT);
_resourceComponent.updateCost(_requirements.purchaseVO.energyCost,
_requirements.purchaseVO.energyAmountShort == 0, CurrencyEnum.ENERGY);
_resourceComponent.updateCost(_requirements.purchaseVO.syntheticCost,
_requirements.purchaseVO.syntheticAmountShort == 0, CurrencyEnum.SYNTHETIC);

switch (_state)
{
case ConstructionView.BUILD:
var
```

```
building:BuildingVO = new BuildingVO();
var action:String = (_prototype is BuildingVO) ? _upgrade : _build;
var instant:String = (_prototype is BuildingVO) ? _upgradeNow : _buildNow;
_actionComponent = new ActionComponent(new ButtonPrototype(action, onActionBtnClick),
new ButtonPrototype(instant, onActionBtnClick), new ButtonPrototype(_getResources,
onClickCannotAffordResourceDialog),
new ButtonPrototype(instant, onActionBtnClick));
_actionComponent.instantCost = _requirements.purchaseVO.premium;

if (_prototype is BuildingVO && _prototype.getValue("upgrade") != "")
building.prototype = presenter.getBuildingUpgrade(_prototype.getValue('upgrade'));
else
building.prototype = _prototype;

_actionComponent.timeCost = building.buildTimeSeconds;
_actionComponent.requirements = _requirements;

var requirement:IRequirement;
for (var i:int = 0; i < _requirements.requirements.length; ++i)
{
requirement = _requirements.requirements[i];
if (!(requirement is CategoryNotBuildingRequirement) && !requirement.isMet)
{
_actionComponent.enabled = false;
break;
}
}
break;

case ConstructionView.RESEARCH:
_actionComponent = new ActionComponent(new ButtonPrototype(_research,
onActionBtnClick), new ButtonPrototype(_researchNow, onActionBtnClick), new
ButtonPrototype(_getResources, onClickCannotAffordResourceDialog),
new ButtonPrototype(_researchNow, onActionBtnClick));
_actionComponent.timeCost = _prototype.buildTimeSeconds;
_actionComponent.enabled = _requirements.allMet;

//see if this is a blueprint
_blueprint = presenter.getBlueprint(_prototype.name);
if (_blueprint)
{
_actionComponent.visible = false;
_blueprintActionComponent = new BlueprintActionComponent(
new ButtonPrototype(_completeBtnText, onBlueprintFullPurchase), new
ButtonPrototype(_purchaseOneText, onBlueprintPartialPurchase),
new ButtonPrototype(_purchaseOneText, popPaywall), new ButtonPrototype(_completeBtnText,
popPaywall));
_blueprintActionComponent.x = _resourceComponent.x + _resourceComponent.width + 25;
_blueprintActionComponent.y = _resourceComponent.y + 3;
_blueprintActionComponent.fullCost = presenter.getBlueprintHardCurrencyCost(_blueprint,
_blueprint.partsRemaining);
```

```
_blueprintActionComponent.partialCost = presenter.getBlueprintHardCurrencyCost(_blueprint,
1);
if (_blueprint.partsCompleted == _blueprint.totalParts || _blueprint.partsCollected >=
_blueprint.totalParts)
_blueprintActionComponent.visible = false;
_resourceComponent.visible = false;
}

if(CONFIG::IS_CRYPTO)
{
if (_blueprint)
{
_actionComponent.visible = false;
_blueprintActionComponent_complete = new BlueprintActionComponent(
new ButtonPrototype(_completeResearchBtnText, onBlueprintResearchComplete), new
ButtonPrototype(_purchaseOneText, onBlueprintPartialPurchase),
new ButtonPrototype(_purchaseOneText, popPaywall), new
ButtonPrototype(_completeResearchBtnText, popPaywall));
_blueprintActionComponent_complete.x = _resourceComponent.x + _resourceComponent.width
+ 25;
_blueprintActionComponent_complete.y = _resourceComponent.y + 3;
_blueprintActionComponent_complete.fullCost = 0;
_blueprintActionComponent_complete.partialCost = 0;
if (_blueprint.partsCompleted == 0 && _blueprint.partsCollected >= _blueprint.totalParts)
_blueprintActionComponent_complete.visible = true;
else
_blueprintActionComponent_complete.visible = false;
_resourceComponent.visible = false;
}
}

break;
}
_actionComponent.instantCost = _requirements.purchaseVO.premium;
_actionComponent.requirements = _requirements;
_actionComponent.x = _resourceComponent.x + _resourceComponent.width + 50;
_actionComponent.y = _resourceComponent.y + 3;

//show the requirements
var currentLabel:RequirementLabel;
var yPos:int = 0;
for (i = 0; i < _requirements.requirements.length; i++)
{
var hText:String = _requirements.requirementsToHtml(_requirements.requirements[i]);

if (hText != "")
{
currentLabel = new RequirementLabel(0, 0, 12, 0xF0F0F0, 275, 30, true, 1);
currentLabel.lbl.multiline = true;
currentLabel.lbl.constrictTextToSize
```

```
= false;
currentLabel.lbl.align = TextFormatAlign.LEFT;
currentLabel.lbl.htmlText = hText;
currentLabel.x = _requirementsBG.x + 6 + (325 * Math.floor(_requirementLabels.length / 3));
currentLabel.y = _requirementsBG.y + 2 + yPos;
yPos += currentLabel.lbl.textHeight + 1;
if (_requirementLabels.length == 2)
yPos = 0;
currentLabel.formatObject();

if (!(_requirements.requirements[i].isMet))
{
currentLabel.checkMark.visible = false;
currentLabel.showLink = true;
} else
{
currentLabel.checkMark.visible = true;
currentLabel.showLink = false;
}
_requirementLabels.push(currentLabel);
}
}
}

private function onClickLink( event:TextEvent ):void
{
var proto:IPrototype = presenter.getPrototypeByName(event.text);
if (proto)
{
var key:uint = uint(proto.getValue('type'));
var isBuilding:Boolean;
switch (key)
{
case TypeEnum.BUILDING_TT:
case TypeEnum.BASE_SHIELD_TT:
case TypeEnum.BASE_TURRET_TT:
isBuilding = true;
}

var view:ConstructionInfoView =
ConstructionInfoView(_viewFactory.createView(ConstructionInfoView));
if (isBuilding)
{
var building:BuildingVO = presenter.getBuildingVOByClass(proto.itemClass);
view.setup(ConstructionView.BUILD, building ? building : proto);
} else
view.setup(ConstructionView.RESEARCH, proto);
_viewFactory.notify(view);
}
destroy();
}
```

```
private function popBusyDialog():void
{
var title:String = _buildProjectTitle;
var body:String = _buildProjectAlertBody;

var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_speedUpBtnText, speedUpTransaction,
[presenter.getStarbaseBuildingTransaction(_prototype.getValue("constructionCategory"))], true,
ButtonEnum.GOLD_A));
buttons.push(new ButtonPrototype(_cancelBtnText));
showConfirmation(title, body, buttons);
}

private function onActionBtnClick( e:MouseEvent = null ):void
{
if(_actionComponent == null)
return;

if(_prototype == null)
return;

var purchaseType:int = (!e || e.currentTarget == _actionComponent.actionBtn) ?
PurchaseTypeEnum.NORMAL : PurchaseTypeEnum.INSTANT;
if (purchaseType == PurchaseTypeEnum.NORMAL && _requirements &&
_requirements.purchaseVO && !_requirements.purchaseVO.canPurchase)
purchaseType = PurchaseTypeEnum.GET_RESOURCES;
switch (_state)
{
case ConstructionView.BUILD:
if (presenter.getStarbaseBuildingTransaction(_prototype.getValue("constructionCategory")) !=
null)
popBusyDialog();
else
{
if (purchaseType == PurchaseTypeEnum.INSTANT && _requirements &&
_requirements.purchaseVO && !_requirements.purchaseVO.canPurchaseWithPremium)
{
popPaywall(null);
return;
}
if (_prototype is BuildingVO)
presenter.performTransaction(TransactionEvent.STARBASE_BUILDING_UPGRADE,
BuildingVO(_prototype), purchaseType);
else
presenter.performTransaction(TransactionEvent.STARBASE_BUILDING_BUILD, _prototype,
purchaseType);
if (_callback != null)
_callback();
destroy();
```

```
}
break;

case ConstructionView.RESEARCH:
if (presenter.getStarbaseResearchTransaction(_prototype.getValue("requiredBuildingClass")) !=
null)
popBusyDialog();
else
{
if (purchaseType == PurchaseTypeEnum.INSTANT && _requirements &&
_requirements.purchaseVO && !_requirements.purchaseVO.canPurchaseWithPremium)
{
popPaywall(null);
return;
}
presenter.performTransaction(TransactionEvent.STARBASE_RESEARCH, _prototype,
purchaseType);
if (_callback != null)
_callback();
destroy();
}
break;
}
}

private function onClickCannotAffordResourceDialog( e:MouseEvent ):void
{
switch (_state)
{
case ConstructionView.BUILD:
if (presenter.getStarbaseBuildingTransaction(_prototype.getValue("constructionCategory")) !=
null)
{
popBusyDialog();
return;
}
break;

case ConstructionView.RESEARCH:
if (presenter.getStarbaseResearchTransaction(_prototype.getValue("requiredBuildingClass")) !=
null)
{
popBusyDialog();
return;
}
break;
}

if (_requirements.purchaseVO.canPurchaseResourcesWithPremium)
{
```

```
var purchaseVO:PurchaseVO = _requirements.purchaseVO;
var view:ResourceModalView =
ResourceModalView(_viewFactory.createView(ResourceModalView));
_viewFactory.notify(view);
view.setUp(purchaseVO.creditsAmountShort, purchaseVO.alloyAmountShort,
purchaseVO.energyAmountShort, purchaseVO.syntheticAmountShort,
'CodeString.Alert.BuyResources.Title', 'CodeString.Alert.BuyResources.Body',
false, onActionBtnClick, purchaseVO.resourcePremiumCost);
} else
popPaywall();
}

private function onSpecialClicked( e:MouseEvent ):void
{
var selectedAsset:String = _prototype.asset;
var selectedLevel:int = _prototype.getValue('level');
var creditRefund:int = _prototype.creditsCost;
var alloyRefund:int = _prototype.alloyCost;
var energyRefund:int = _prototype.energyCost;
var syntheticRefund:int = _prototype.syntheticCost;

/*var allBuildings:Vector.<IPrototype> = presenter.buildingPrototypes;
var len:uint = allBuildings.length;
var currentVO:IPrototype;
for (var i:uint = 0; i < len; ++i)
{
currentVO = allBuildings[i];
if (currentVO != null && selectedAsset == currentVO.asset && selectedLevel >
currentVO.getValue('level'))
{
creditRefund = _buildingVO.creditsCost;
alloyRefund = _buildingVO.alloyCost;
energyRefund = _buildingVO.energyCost;
syntheticRefund = _buildingVO.syntheticCost;
}
}*/

creditRefund = _prototype.creditsCost;
alloyRefund = _prototype.alloyCost;
energyRefund = _prototype.energyCost;
syntheticRefund = _prototype.syntheticCost;

var view:ResourceModalView =
ResourceModalView(_viewFactory.createView(ResourceModalView));
_viewFactory.notify(view);
view.setUp(Math.floor(creditRefund * 0.20), Math.floor(alloyRefund * 0.20),
Math.floor(syntheticRefund * 0.20), Math.floor(energyRefund * 0.20),
'CodeString.BuildRecycle.Title.Recycle', 'CodeString.BuildRecycle.Refund',
true, onRecycleClick);
}
```

```
private function onBlueprintFullPurchase( e:MouseEvent ):void
{
if (_blueprint && !_blueprint.complete)
{
presenter.purchaseBlueprint(_blueprint, _blueprint.partsRemaining);
if (_callback != null)
_callback();
destroy();
}
}

private function onBlueprintPartialPurchase( e:MouseEvent ):void
{
if (_blueprint && !_blueprint.complete)
{
presenter.purchaseBlueprint(_blueprint, 1);
if (_callback != null)
_callback();
destroy();
}
}

private function onBlueprintResearchComplete( e:MouseEvent ):void
{
if (_blueprint && !_blueprint.complete)
{
presenter.completeBlueprintResearch(_blueprint);
if (_callback != null)
_callback();
destroy();
}
}

private function popPaywall( e:MouseEvent = null ):void
{
CommonFunctionUtil.popPaywall();
}

private function onRecycleClick( e:MouseEvent = null ):void
{
presenter.performTransaction(TransactionEvent.STARBASE_BUILDING_RECYCLE,
_prototype, PurchaseTypeEnum.INSTANT);
if (_callback != null)
_callback();
destroy();
}

private function showFullTooltip( e:MouseEvent ):void
{
```

```
_fullStatTooltipComponent.layoutTooltip(_statWindowString, 25, 68, 10, 6);

var view:StatInformationView =
StatInformationView(_viewFactory.createView(StatInformationView));
view.SetUp(_fullStatTooltipComponent);
_viewFactory.notify(view);
}
private function playVOSound( e:MouseEvent ):void
{
if (_soundToPlay != null)
presenter.playSound(_soundToPlay);
}

protected function speedUpTransaction( transaction:TransactionVO ):void
{
if (transaction)
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
nStoreView.setSelectedTransaction(transaction);
}
}

public function set callback( v:Function ):void
{
_callback = v;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IConstructionPresenter ):void
{
_presenter = v;
}

public function get presenter():IConstructionPresenter
{
return IConstructionPresenter(_presenter);
}

override public function destroy():void
{
super.destroy();

ObjectPool.give(_bg);
_bg = null;
_closeButton
```

```
= UIFactory.destroyButton(_closeButton);
_description = UIFactory.destroyLabel(_description);
ObjectPool.give(_image);
_image = null;
_imageFrame = UIFactory.destroyPanel(_imageFrame);
_infoPanel = UIFactory.destroyPanel(_infoPanel);
_prototype = null;
_requirements = null;
_specialButton = UIFactory.destroyButton(_specialButton);
_statsPanel = UIFactory.destroyPanel(_statsPanel);
_title = UIFactory.destroyLabel(_title);
ObjectPool.give(_tooltipComponent);
_tooltipComponent = null;
ObjectPool.give(_fullStatTooltipComponent);
_fullStatTooltipComponent = null;

if (_requirementsPanel)
{
_actionComponent = null;
_blueprint = null;
_blueprintActionComponent = null;
_blueprintActionComponent_complete = null;
_requirementsPanel = UIFactory.destroyPanel(_requirementsPanel);
ObjectPool.give(_resourceComponent);
_resourceComponent = null;
_requirementsBG = UIFactory.destroyPanel(_requirementsBG);
for (var i:int = 0; i < _requirementLabels.length; i++)
_requirementLabels[i].lbl.removeEventListener(TextEvent.LINK, onClickLink);
_requirementLabels.length = 0;
_requirementLabels = null;
}

_infoBtn.destroy();
_infoBtn = null;
if(_soundBtn != null)
{
_soundBtn.destroy();
_soundBtn = null;
}
_callback = null;
}
}
}

-------------------------------------
File 799: igw\com\ui\modal\construction\ConstructionItem.as
package com.ui.modal.construction
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import
```

```
com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.blueprint.BlueprintVO;
import com.model.prototype.IPrototype;
import com.presenter.starbase.IConstructionPresenter;
import com.service.language.Localization;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.misc.TooltipComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.filters.GlowFilter;
import flash.text.TextFormatAlign;

import org.adobe.utils.StringUtil;
import org.shared.ObjectPool;

public class ConstructionItem extends Sprite
{
private var _actionButton:BitmapButton;
private var _bg:ScaleBitmap;
private var _description:Label;
private var _image:ImageComponent;
private var _imageFrame:ScaleBitmap;
private var _lock:Bitmap;
private var _presenter:IConstructionPresenter;
private var _prototype:IPrototype;
private var _state:int;
private var _statsBG:ScaleBitmap;
private var _title:Label;
private var _tooltipComponent:TooltipComponent;
private var _tooltips:Tooltips;

private var _buildText:String = 'CodeString.ConstructionItem.Build'; //BUILD
private var _upgradeText:String = 'CodeString.ConstructionItem.Upgrade'; //UPGRADE
private var _addText:String = 'CodeString.ConstructionItem.Add'; //ADD
private var _detailsText:String = 'CodeString.ConstructionItem.Details'; //DETAILS
private var _researchText:String = 'CodeString.ConstructionItem.Research'; //RESEARCH
private var _completeText:String = 'CodeString.ConstructionItem.Complete'; //COMPLETE
private var _lockedText:String = 'CodeString.ConstructionItem.Locked'; //LOCKED

public function init( prototype:IPrototype, presenter:IConstructionPresenter, state:int,
tooltips:Tooltips ):void
{
_presenter
```

```
= presenter;
_prototype = prototype;
_state = state;
_tooltips = tooltips;
_bg = UIFactory.getPanel(PanelEnum.CONTAINER_INNER, 600, 118);

_image = ObjectPool.get(ImageComponent);
_image.init(100, 100);
_image.center = true;
_image.x = _image.y = 9;
_image.mouseEnabled = _image.mouseChildren = false;

_imageFrame = UIFactory.getPanel(PanelEnum.CHARACTER_FRAME, 100, 100, 9, 9);

_lock = UIFactory.getBitmap("IconBlueLockedBMD");
_lock.x = _imageFrame.x + (100 - _lock.width) * .5;
_lock.y = _imageFrame.y + (100 - _lock.height) * .5;
_lock.visible = false;

_title = UIFactory.getLabel(LabelEnum.TITLE, 300, 45, _image.x + 108, 8);
_title.textColor = 0xfbefaf;
_title.align = TextFormatAlign.LEFT;
_title.useLocalization = false;
_title.bold = false;
_title.constrictTextToSize = true;
_title.mouseEnabled = false;

_statsBG = UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL, 474, 61,
_title.x, 49);

_tooltipComponent = ObjectPool.get(TooltipComponent);
_tooltipComponent.init(2, 474);
_tooltipComponent.x = _statsBG.x;
_tooltipComponent.y = _statsBG.y;
_tooltipComponent.mouseEnabled = _tooltipComponent.mouseChildren = false;

finalize();

addChild(_bg);
addChild(_actionButton);
if (_description)
addChild(_description);
addChild(_image);
addChild(_imageFrame);
addChild(_lock);
addChild(_statsBG);
addChild(_title);
addChild(_tooltipComponent);
}

private
```

```
function finalize():void
{
var glow:GlowFilter;
var rarity:String;

//load the image and display the title
var assetVO:AssetVO = _presenter.getAssetVO(_prototype);

if(assetVO)
{
_presenter.loadImage(assetVO.mediumImage, _image.onImageLoaded);
_title.text = Localization.instance.getString(assetVO.visibleName).toUpperCase();
}

var proto:IPrototype;
switch (_state)
{
case ConstructionView.BUILD:
var count:int = _presenter.getBuildingCount(_prototype.itemClass);
var maxCount:int = _presenter.getBuildingMaxCount(_prototype.itemClass);
if (count < maxCount)
_actionButton = UIFactory.getButton(ButtonEnum.GREEN_A, 138, 31, 452, 8, _buildText);
else if (maxCount == 1 && count == 1 &&
_presenter.getBuildingVOByClass(_prototype.itemClass, true).level != 10)
_actionButton = UIFactory.getButton(ButtonEnum.GREEN_A, 138, 31, 452, 8, _upgradeText);
else
_actionButton = UIFactory.getButton(ButtonEnum.BLUE_A, 138, 31, 452, 8, _detailsText);
_title.text += " " + count + "/" + maxCount;
_description = UIFactory.getLabel(LabelEnum.DESCRIPTION, _statsBG.width - 4,
_statsBG.height - 4, _statsBG.x + 2, _statsBG.y + 2);

if(assetVO)
_description.text = assetVO.descriptionText;

_tooltips.addTooltip(this, null, null, StringUtil.getTooltip(_prototype.getValue("type"), _prototype,
false, null));
break;

case ConstructionView.COMPONENT:
_actionButton = UIFactory.getButton(ButtonEnum.BLUE_A, 138, 31, 452, 8, _addText);
_tooltipComponent.layoutTooltip(StringUtil.getTooltip(_prototype.getValue("type"), _prototype,
false, null, true));
_tooltips.addTooltip(this, null, null, StringUtil.getTooltip(_prototype.getValue("type"), _prototype,
false, null));

rarity = _prototype.getUnsafeValue('rarity');
if (rarity != 'Common')
{
glow = CommonFunctionUtil.getRarityGlow(rarity);
_title.textColor = glow.color;
_imageFrame.filters
```

```
= [glow];
}
break;

case ConstructionView.RESEARCH:
var blueprint:BlueprintVO = _presenter.getBlueprint(_prototype.name);
var requirementMet:Boolean = _presenter.requirementsMet(_prototype);
if (_presenter.isResearched(_prototype.name))
_actionButton = UIFactory.getButton(ButtonEnum.BLUE_A, 138, 31, 452, 8, _detailsText);
else if (requirementMet)
_actionButton = UIFactory.getButton(ButtonEnum.GREEN_A, 138, 31, 452, 8, _researchText);
else if (!requirementMet)
{
if (blueprint && blueprint.partsCompleted < blueprint.totalParts)
_actionButton = UIFactory.getButton(ButtonEnum.GOLD_A, 138, 31, 452, 8, _completeText);
else
_actionButton = UIFactory.getButton(ButtonEnum.RED_A, 138, 31, 452, 8, _lockedText);
_lock.visible = true;
_image.filters = [CommonFunctionUtil.getGreyScaleFilter()];
}

rarity = _prototype.getUnsafeValue('rarity');
if (rarity != 'Common')
{
glow = CommonFunctionUtil.getRarityGlow(rarity);
_title.textColor = glow.color;
_imageFrame.filters = [glow];
} else
_imageFrame.filters = [];

//see if this is a blueprint
if (blueprint && (CONFIG::IS_CRYPTO || blueprint.partsCollected < blueprint.totalParts))
_title.text += " " + blueprint.partsCollected + "/" + blueprint.totalParts;


proto = _presenter.getResearchItemPrototypeByName(_prototype.getValue("referenceName"));
_tooltipComponent.layoutTooltip(StringUtil.getTooltip(proto.getValue("type"), proto, false, null,
true));
_tooltips.addTooltip(this, null, null, StringUtil.getTooltip(proto.getValue("type"), proto, false,
null));
break;
}

_actionButton.hitArea = this;
}

public function get prototype():IPrototype { return _prototype; }

public function destroy():void
{
while
```

```
(numChildren > 0)
removeChildAt(0);

_actionButton = UIFactory.destroyButton(_actionButton);
_bg = UIFactory.destroyPanel(_bg);
if (_description)
_description = UIFactory.destroyLabel(_description);
ObjectPool.give(_image);
_image = null;
_imageFrame = UIFactory.destroyPanel(_imageFrame);
_lock = UIFactory.destroyPanel(_lock);
_presenter = null;
_prototype = null;
_statsBG = UIFactory.destroyPanel(_statsBG);
_title = UIFactory.destroyLabel(_title);
ObjectPool.give(_tooltipComponent);
_tooltipComponent = null;
_tooltips.removeTooltip(this);
_tooltips = null;
}
}
}

-----------------------------------
File 800: igw\com\ui\modal\construction\ConstructionList.as
package com.ui.modal.construction
{
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.prototype.IPrototype;
import com.presenter.starbase.IConstructionPresenter;
import com.ui.UIFactory;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.building.RefitBuildingView;
import com.ui.modal.shipyard.ShipyardView;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.osflash.signals.Signal;
import org.parade.core.IView;
import org.parade.core.IViewFactory;
import org.shared.ObjectPool;

public class ConstructionList extends Sprite
{
private var _bg:Sprite;
private
```

```
var _closeSignal:Signal;
private var _holder:Sprite;
private var _items:Vector.<ConstructionItem>;
private var _maxHeight:int;
private var _presenter:IConstructionPresenter;
private var _scrollBar:VScrollbar;
private var _scrollRect:Rectangle;
private var _state:int;
private var _tooltips:Tooltips;
private var _viewFactory:IViewFactory;

internal function init( presenter:IConstructionPresenter, state:int, tooltips:Tooltips,
viewFactory:IViewFactory ):void
{
_presenter = presenter;
_state = state;
_tooltips = tooltips;
_bg = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED,
PanelEnum.HEADER_NOTCHED_RIGHT, 626, 494, 30, 0, 0, "TEST", LabelEnum.H3);
_closeSignal = new Signal();
_viewFactory = viewFactory;

_holder = new Sprite();
_holder.x = 4;
_holder.y = 34;

_items = new Vector.<ConstructionItem>;
_maxHeight = 0;

_scrollRect = new Rectangle(0, 0, 626, 488);

//scrollbar
_scrollBar = new VScrollbar();
_scrollBar.init(7, _scrollRect.height - 15, 606, 32, new Rectangle(0, 4, 5, 3), '', 'ScrollBarBMD', '',
false, this);
_scrollBar.onScrollSignal.add(onChangedScroll);
_scrollBar.updateDisplayedHeight(_scrollRect.height);
_scrollBar.updateScrollableHeight(_maxHeight);
_scrollBar.maxScroll = 29.5;

addChild(_bg);
addChild(_holder);
addChild(_scrollBar);
}

internal function update( items:Vector.<IPrototype> ):void
{
clearCurrentItems();

var item:ConstructionItem;
for
```

```
(var i:int = 0; i < items.length; i++)
{
item = ObjectPool.get(ConstructionItem);
item.init(items[i], _presenter, _state, _tooltips);
item.y = _maxHeight;
item.addEventListener(MouseEvent.CLICK, onItemClicked, false, 0, true);
_maxHeight += 122;
_holder.addChild(item);
_items.push(item);
}

_scrollBar.updateScrollableHeight(_maxHeight);
_scrollBar.updateDisplayedHeight(_scrollRect.height);
_scrollBar.updateScrollY(0);
}

internal function onSpecialButtonClicked( e:MouseEvent ):void
{
switch (_state)
{
case ConstructionView.COMPONENT:
var targetView:IView = _presenter.getView(ShipyardView);
if (targetView)
ShipyardView(targetView).onComponentSelected(null);
else
{
targetView = _presenter.getView(RefitBuildingView);
if (targetView)
RefitBuildingView(targetView).onModuleSelected(null);
}
onClose();
break;
}
}

private function onItemClicked( e:MouseEvent ):void
{
var item:ConstructionItem = ConstructionItem(e.currentTarget);
var view:IView;
switch (_state)
{
case ConstructionView.BUILD:
view = _viewFactory.createView(ConstructionInfoView);
ConstructionInfoView(view).setup(_state, item.prototype);
ConstructionInfoView(view).callback = onClose;
break;

case ConstructionView.COMPONENT:
var targetView:IView = _presenter.getView(ShipyardView);
if (targetView)
ShipyardView(targetView).onComponentSelected(item.prototype);
```

```
else
{
targetView = _presenter.getView(RefitBuildingView);
if (targetView)
RefitBuildingView(targetView).onModuleSelected(item.prototype);
}
onClose();
break;

case ConstructionView.RESEARCH:
view = _viewFactory.createView(ConstructionInfoView);
ConstructionInfoView(view).setup(_state, item.prototype);
ConstructionInfoView(view).callback = onClose;
break;
}

if (view)
_viewFactory.notify(view);
}

private function clearCurrentItems():void
{
if (_items)
{
for (var i:int = 0; i < _items.length; i++)
{
_holder.removeChild(_items[i]);
_items[i].removeEventListener(MouseEvent.CLICK, onItemClicked);
ObjectPool.give(_items[i]);
}
_items.length = _maxHeight = 0;
}
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

private function onClose():void
{
_closeSignal.dispatch();
}

internal function addCloseListener( listener:Function ):void { _closeSignal.add(listener); }

internal function set title( v:String ):void { Label(_bg.getChildAt(2)).text = v; }

public
```

```
function destroy():void
{
while (numChildren > 0)
removeChildAt(0);

clearCurrentItems();
_bg = UIFactory.destroyPanel(_bg);
_closeSignal.removeAll();
_closeSignal = null;
_holder = null;
_presenter = null;
_scrollBar.destroy();
_scrollBar = null;
_scrollRect = null;
_tooltips.removeTooltip(null, this);
_tooltips = null;
_viewFactory = null;
}
}
}

------------------------------------
File 801: igw\com\ui\modal\construction\ConstructionView.as
package com.ui.modal.construction
{
import com.enum.FilterEnum;
import com.enum.SlotComponentEnum;
import com.enum.StarbaseCategoryEnum;
import com.enum.TypeEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.IConstructionPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.accordian.AccordianComponent;
import com.ui.core.component.accordian.AccordianGroup;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.tooltips.Tooltips;

import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

import org.shared.ObjectPool;

import com.util.CommonFunctionUtil;

/**
*
```

```
ConstructionView is a multipurpose view that handles building, research and component
selection.
* To use this view, create a new instance and then call the openOn method passing in the state
* (BUILD, COMPONENT, or RESEARCH) and the groupID and subItemID for the accordian.
*/
public class ConstructionView extends View
{
public static const BUILD:int = 0;
public static const COMPONENT:int = 1;
public static const RESEARCH:int = 2;

private static const LAST_VIEWED:Dictionary = new Dictionary();

private var _accordian:AccordianComponent;
private var _bg:DefaultWindowBG;
private var _closeButton:BitmapButton;
private var _componentSlot:String;
private var _groupID:String;
private var _highestButton:BitmapButton;
private var _advancedButton:BitmapButton;
private var _commonButton:BitmapButton;
private var _uncommonButton:BitmapButton;
private var _rareButton:BitmapButton;
private var _epicButton:BitmapButton;
private var _legendaryButton:BitmapButton;
private var _list:ConstructionList;
private var _specialButton:BitmapButton;
private var _state:int;
private var _subItemID:String;
private var _tooltips:Tooltips;

//localized code strings
private var _infrastructureButton:String = 'CodeString.Build.InfrastructureBtn';
private var _defenseButton:String = 'CodeString.Build.DefenseBtn';
private var _researchButton:String = 'CodeString.Build.ResearchBtn';
private var _fleetsButton:String = 'CodeString.Build.FleetsBtn';
private var _starbaseStructureButton:String = 'CodeString.Build.StarbaseStructureBtn';
private var _clearSlot:String = "CodeString.Shared.ClearSlot";
private var _showHighest:String = "CodeString.Shared.ShowHighest";
private var _showAdvancedOnly:String = "CodeString.Shared.ShowAdvancedOnly";
private var _showCommonOnly:String = "CodeString.Shared.ShowCommonOnly";
private var _showUncommonOnly:String = "CodeString.Shared.ShowUncommonOnly";
private var _showRareOnly:String = "CodeString.Shared.ShowRareOnly";
private var _showEpicOnly:String = "CodeString.Shared.ShowEpicOnly";
private var _showLegendaryOnly:String = "CodeString.Shared.ShowLegendaryOnly";
private var _construction:String = 'CodeString.BuildInformation.Title.Construction';
//CONSTRUCTION

private var _arcWeapons:String = 'CodeString.ComponentSelection.Title.ArcWeapons'; //ARC
WEAPONS
private
```

```
var _spinalWeapons:String = 'CodeString.ComponentSelection.Title.SpinalWeapons'; //SPINAL
WEAPONS
private var _weapons:String = 'CodeString.ComponentSelection.Title.Weapons'; //WEAPONS
private var _technology:String = 'CodeString.ComponentSelection.Title.Technology';
//TECHNOLOGY
private var _defense:String = 'CodeString.ComponentSelection.Title.Defense'; //DEFENSE
private var _structure:String = 'CodeString.ComponentSelection.Title.Structure'; //STRUCTURE
private var _baseTurrets:String = 'CodeString.ComponentSelection.Title.BaseTurrets';
//TURRETS WEAPONS
private var _baseShields:String = 'CodeString.ComponentSelection.Title.BaseShields';
//SHIELDS
private var _droneBay:String = 'CodeString.ModuleClass.DroneBay'; //Drone Bay

private var _blueprints:String = 'CodeString.Research.Blueprints';
private var _research:String = 'CodeString.Research.Title';

private var _componentText:String = 'CodeString.ConstructionView.Component';
//COMPONENT
private var _closeText:String = 'CodeString.ConstructionView.Close'; //CLOSE
private var _defenseText:String = 'CodeString.ConstructionView.Defense'; //DEFENSE
private var _hullsText:String = 'CodeString.ConstructionView.Hulls'; //HULLS
private var _techText:String = 'CodeString.ConstructionView.Tech'; //TECH
private var _weaponsText:String = 'CodeString.ConstructionView.Weapons'; //WEAPONS


[Inject]
override public function init():void
{
super.init();
presenter.addOnTransactionRemovedListener(onTransactionChanged);
_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(894, 535);

_accordian = ObjectPool.get(AccordianComponent);
_accordian.init(244, 52);
_accordian.x = _bg.bg.x + 14;
_accordian.y = _bg.bg.y + 5;
_accordian.addListener(onAccordianSelected);

_closeButton = UIFactory.getButton(ButtonEnum.BLUE_A, 240, 40, _bg.width - 263, _bg.height
+ 9, _closeText);

var rarityButtonOffset:int = 60;
var rarityButtonInitialOffset:int = 130;
var rarityButtonWidth:int = 60;

_highestButton = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0, _showHighest,
LabelEnum.DEFAULT_OPEN_SANS);
_highestButton.label.setSize(100, 25);
_highestButton.label.align = TextFormatAlign.RIGHT;
_highestButton.label.x
```

```
-= _highestButton.width - 15;
_highestButton.label.y -= 4;
_highestButton.x = _bg.x + _bg.width - 30;
_highestButton.y = _accordian.y + 4;
_highestButton.selected = true;

_legendaryButton = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0,
_showLegendaryOnly, LabelEnum.DEFAULT_OPEN_SANS);
_legendaryButton.label.setSize(rarityButtonWidth, 25);
_legendaryButton.label.textColor = CommonFunctionUtil.getRarityColor('Legendary');
_legendaryButton.label.align = TextFormatAlign.LEFT;
_legendaryButton.label.x -= _legendaryButton.width - 15;
_legendaryButton.label.y -= 4;
_legendaryButton.x = _bg.x + _bg.width - rarityButtonOffset - rarityButtonInitialOffset;
_legendaryButton.y = _accordian.y + 4;
_legendaryButton.selected = false;

_commonButton = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0,
_showCommonOnly, LabelEnum.DEFAULT_OPEN_SANS);
_commonButton.label.setSize(rarityButtonWidth, 25);
_commonButton.label.textColor = CommonFunctionUtil.getRarityColor('Common');
_commonButton.label.align = TextFormatAlign.LEFT;
_commonButton.label.x -= _commonButton.width - 15;
_commonButton.label.y -= 4;
_commonButton.x = _bg.x + _bg.width - 6*rarityButtonOffset - rarityButtonInitialOffset;
_commonButton.y = _accordian.y + 4;
_commonButton.selected = false;

_advancedButton = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0,
_showAdvancedOnly, LabelEnum.DEFAULT_OPEN_SANS);
_advancedButton.label.setSize(rarityButtonWidth, 25);
_advancedButton.label.textColor = CommonFunctionUtil.getRarityColor('Advanced1');
_advancedButton.label.align = TextFormatAlign.LEFT;
_advancedButton.label.x -= _advancedButton.width - 15;
_advancedButton.label.y -= 4;
_advancedButton.x = _bg.x + _bg.width - 5*rarityButtonOffset - rarityButtonInitialOffset;
_advancedButton.y = _accordian.y + 4;
_advancedButton.selected = false;

_uncommonButton = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0,
_showUncommonOnly, LabelEnum.DEFAULT_OPEN_SANS);
_uncommonButton.label.setSize(rarityButtonWidth, 25);
_uncommonButton.label.textColor = CommonFunctionUtil.getRarityColor('Uncommon');
_uncommonButton.label.align = TextFormatAlign.LEFT;
_uncommonButton.label.x -= _uncommonButton.width - 15;
_uncommonButton.label.y -= 4;
_uncommonButton.x = _bg.x + _bg.width - 4*rarityButtonOffset - rarityButtonInitialOffset;
_uncommonButton.y = _accordian.y + 4;
_uncommonButton.selected = false;

_rareButton
```

```
= UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0, _showRareOnly,
LabelEnum.DEFAULT_OPEN_SANS);
_rareButton.label.setSize(rarityButtonWidth, 25);
_rareButton.label.textColor = CommonFunctionUtil.getRarityColor('Rare');
_rareButton.label.align = TextFormatAlign.LEFT;
_rareButton.label.x -= _rareButton.width - 15;
_rareButton.label.y -= 4;
_rareButton.x = _bg.x + _bg.width - 3*rarityButtonOffset - rarityButtonInitialOffset;
_rareButton.y = _accordian.y + 4;
_rareButton.selected = false;

_epicButton = UIFactory.getButton(ButtonEnum.CHECKBOX, 0, 0, 0, 0, _showEpicOnly,
LabelEnum.DEFAULT_OPEN_SANS);
_epicButton.label.setSize(rarityButtonWidth, 25);
_epicButton.label.textColor = CommonFunctionUtil.getRarityColor('Epic');
_epicButton.label.align = TextFormatAlign.LEFT;
_epicButton.label.x -= _epicButton.width - 15;
_epicButton.label.y -= 4;
_epicButton.x = _bg.x + _bg.width - 2*rarityButtonOffset - rarityButtonInitialOffset;
_epicButton.y = _accordian.y + 4;
_epicButton.selected = false;

_list = ObjectPool.get(ConstructionList);
_list.init(presenter, _state, _tooltips, _viewFactory);
_list.x = _accordian.x + 248;
_list.y = _accordian.y;
_list.addCloseListener(onClose);

addChild(_bg);
addChild(_closeButton);
addChild(_list);
addChild(_accordian);
addChild(_highestButton);
addChild(_advancedButton);
addChild(_commonButton);
addChild(_uncommonButton);
addChild(_rareButton);
addChild(_epicButton);
addChild(_legendaryButton);

addListener(_bg.closeButton, MouseEvent.CLICK, onClose);
addListener(_closeButton, MouseEvent.CLICK, onClose);
addListener(_highestButton, MouseEvent.CLICK, onHighestSelected);
addListener(_commonButton, MouseEvent.CLICK, onCommonOnlySelected);
addListener(_advancedButton, MouseEvent.CLICK, onAdvancedOnlySelected);
addListener(_uncommonButton, MouseEvent.CLICK, onUncommonOnlySelected);
addListener(_rareButton, MouseEvent.CLICK, onRareOnlySelected);
addListener(_epicButton, MouseEvent.CLICK, onEpicOnlySelected);
addListener(_legendaryButton, MouseEvent.CLICK, onLegendaryOnlySelected);

showState();
```

```
addEffects();
effectsIN();
}
/**
* Must be called before the view is passed to ViewFactory to be shown.
* This method defines the state of the view and specifies which group and
* subItem the accordian should focus on.
* @param state BUILD, COMPONENT or RESEARCH
* @param groupID The group that the accordian should default to
* @param subItemID The subItem that the accordian should default to
*/
public function openOn( state:int, groupID:String, subItemID:String ):void
{
_groupID = groupID;
_state = state;
if (_state == COMPONENT)
{
_componentSlot = subItemID;
subItemID = null;
} else
_subItemID = subItemID;
}

private function onAccordianSelected( groupID:String, subItemID:String, data:* ):void
{
var group:AccordianGroup = _accordian.getGroup(groupID);
if (!subItemID && group.hasSubItems)
{
subItemID = group.subItems[0].id;
_accordian.setSelected(groupID, subItemID)
}
_list.title = (group.hasSubItems) ? group.getSubItem(subItemID).text.toUpperCase() :
group.text.toUpperCase();
switch (_state)
{
case COMPONENT:
_list.update(presenter.getComponents(groupID, subItemID, _componentSlot,
_highestButton.selected, _advancedButton.selected,
_commonButton.selected,_uncommonButton.selected,
_rareButton.selected, _epicButton.selected, _legendaryButton.selected));
break;

case RESEARCH:
_list.update(presenter.getResearchPrototypes(groupID, subItemID));
break;

case BUILD:
default:
_list.update(presenter.getBuildingPrototypes(groupID, subItemID));
break;
```

```
}

_groupID = groupID;
_subItemID = subItemID;

//save the last viewed so that it can be reshown later
if (!LAST_VIEWED.hasOwnProperty(_state))
LAST_VIEWED[_state] = {};
LAST_VIEWED[_state].groupID = _groupID;
LAST_VIEWED[_state].subItemID = _subItemID;
LAST_VIEWED[_state].showHighest = _highestButton.selected;
LAST_VIEWED[_state].showCommonOnly = _commonButton.selected;
LAST_VIEWED[_state].showAdvancedOnly = _advancedButton.selected;
LAST_VIEWED[_state].showUncommonOnly = _uncommonButton.selected;
LAST_VIEWED[_state].showRareOnly = _rareButton.selected;
LAST_VIEWED[_state].showEpicOnly = _epicButton.selected;
LAST_VIEWED[_state].showLegendaryOnly = _legendaryButton.selected;
}

private function showState():void
{
_highestButton.visible = false;
_commonButton.visible = false;
_advancedButton.visible = false;
_uncommonButton.visible = false;
_rareButton.visible = false;
_epicButton.visible = false;
_legendaryButton.visible = false;
switch (_state)
{
case COMPONENT:
_bg.addTitle(_componentText, 300);
_highestButton.visible = true;
_commonButton.visible = true;
_advancedButton.visible = true;
_uncommonButton.visible = true;
_rareButton.visible = true;
_epicButton.visible = true;
_legendaryButton.visible = true;
switch (_groupID)
{
case SlotComponentEnum.SLOT_TYPE_ARC:
_bg.addTitle(_arcWeapons, 300);
_accordian.addGroup(_groupID, _arcWeapons);
_accordian.addSubItemToGroup(_groupID, FilterEnum.ARC_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.ARC_WEAPONS), 0);
break;
case SlotComponentEnum.SLOT_TYPE_SPINAL:
_bg.addTitle(_spinalWeapons, 300);
_accordian.addGroup(_groupID,
```

```
_spinalWeapons);
_accordian.addSubItemToGroup(_groupID, FilterEnum.SPINAL_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.SPINAL_WEAPONS), 0);
break;
case SlotComponentEnum.SLOT_TYPE_WEAPON:
_bg.addTitle(_weapons, 300);
_accordian.addGroup(_groupID, _weapons);
_accordian.addSubItemToGroup(_groupID, FilterEnum.PROJECTILE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.PROJECTILE_WEAPONS), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.BEAM_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.BEAM_WEAPONS), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.GUIDED_WEAPON,
presenter.getFilterNameByKey(FilterEnum.GUIDED_WEAPON), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.LEGACY_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_WEAPONS), 0);
break;
case SlotComponentEnum.SLOT_TYPE_TECH:
_bg.addTitle(_technology, 300);
_accordian.addGroup(_groupID, _technology);
_accordian.addSubItemToGroup(_groupID, FilterEnum.SHIP_TECH,
presenter.getFilterNameByKey(FilterEnum.SHIP_TECH), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.WEAPONS_TECH,
presenter.getFilterNameByKey(FilterEnum.WEAPONS_TECH), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.BEAM_TECH,
presenter.getFilterNameByKey(FilterEnum.BEAM_TECH), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.PROJECTILE_TECH,
presenter.getFilterNameByKey(FilterEnum.PROJECTILE_TECH), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.GUIDED_TECH,
presenter.getFilterNameByKey(FilterEnum.GUIDED_TECH), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.LEGACY_TECH,
presenter.getFilterNameByKey(FilterEnum.LEGACY_TECH), 0);
//PR: Disabling Secondary Tech for the release
//_accordian.addSubItemToGroup(_groupID, FilterEnum.SECONDARY_TECH,
presenter.getFilterNameByKey(FilterEnum.SECONDARY_TECH), 0);
break;
case SlotComponentEnum.SLOT_TYPE_DEFENSE:
_bg.addTitle(_defense, 300);
_accordian.addGroup(_groupID, _defense);
_accordian.addSubItemToGroup(_groupID, FilterEnum.ARMOR,
presenter.getFilterNameByKey(FilterEnum.ARMOR), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.SHIP_SHIELDS,
presenter.getFilterNameByKey(FilterEnum.SHIP_SHIELDS), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.ACTIVE_DEFENSES,
presenter.getFilterNameByKey(FilterEnum.ACTIVE_DEFENSES), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.LEGACY_DEFENSE,
presenter.getFilterNameByKey(FilterEnum.LEGACY_DEFENSE), 0);
break;
case SlotComponentEnum.SLOT_TYPE_STRUCTURE:
_bg.addTitle(_structure, 300);
_accordian.addGroup(_groupID, _structure);
_accordian.addSubItemToGroup(_groupID,
```

```
FilterEnum.INTEGRITY_FIELD,
presenter.getFilterNameByKey(FilterEnum.INTEGRITY_FIELD), 0);
break;
case SlotComponentEnum.SLOT_TYPE_TURRET:
_bg.addTitle(_baseTurrets, 300);
_accordian.addGroup(_groupID, _baseTurrets);
_accordian.addSubItemToGroup(_groupID, FilterEnum.BASE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.BASE_WEAPONS), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.LEGACY_BASE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_BASE_WEAPONS), 0);
break;
case SlotComponentEnum.SLOT_TYPE_SHIELD:
_bg.addTitle(_baseShields, 300);
_accordian.addGroup(_groupID, _baseShields);
_accordian.addSubItemToGroup(_groupID, FilterEnum.BASE_SHIELDS,
presenter.getFilterNameByKey(FilterEnum.BASE_SHIELDS), 0);
_accordian.addSubItemToGroup(_groupID, FilterEnum.LEGACY_BASE_SHIELDS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_BASE_SHIELDS), 0);
break;
case SlotComponentEnum.SLOT_TYPE_DRONE:
_bg.addTitle(_droneBay, 300);
_accordian.addGroup(_groupID, _droneBay);
_accordian.addSubItemToGroup(_groupID, FilterEnum.DRONE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.DRONE_WEAPONS), 0);
break;
}
if (LAST_VIEWED.hasOwnProperty(_state))
{
_highestButton.selected = LAST_VIEWED[_state].showHighest;
_commonButton.selected = LAST_VIEWED[_state].showCommonOnly;
_advancedButton.selected = LAST_VIEWED[_state].showAdvancedOnly;
_uncommonButton.selected = LAST_VIEWED[_state].showUncommonOnly;
_rareButton.selected = LAST_VIEWED[_state].showRareOnly;
_epicButton.selected = LAST_VIEWED[_state].showEpicOnly;
_legendaryButton.selected = LAST_VIEWED[_state].showLegendaryOnly;
_highestButton.selected = LAST_VIEWED[_state].showHighest;
}
_accordian.addSubItemToGroup(_groupID, "Blueprint", _blueprints, 0);
onAccordianSelected(_groupID, null, null);

_specialButton = UIFactory.getButton(ButtonEnum.RED_A, 240, 40, _closeButton.x - 250,
_closeButton.y, _clearSlot);
break;

case RESEARCH:
_bg.addTitle(_research, 300);
_accordian.addGroup(TypeEnum.DEFENSE_DESIGN, _defenseText);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN, FilterEnum.ARMOR,
presenter.getFilterNameByKey(FilterEnum.ARMOR), 0);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN, FilterEnum.SHIP_SHIELDS,
presenter.getFilterNameByKey(FilterEnum.SHIP_SHIELDS),
```

```
0);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN,
FilterEnum.ACTIVE_DEFENSES,
presenter.getFilterNameByKey(FilterEnum.ACTIVE_DEFENSES), 0);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN, FilterEnum.BASE_SHIELDS,
presenter.getFilterNameByKey(FilterEnum.BASE_SHIELDS), 0);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN,
FilterEnum.LEGACY_DEFENSE,
presenter.getFilterNameByKey(FilterEnum.LEGACY_DEFENSE), 0);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN,
FilterEnum.LEGACY_BASE_SHIELDS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_BASE_SHIELDS), 0);
_accordian.addSubItemToGroup(TypeEnum.DEFENSE_DESIGN, "Blueprint", _blueprints, 0);
_accordian.addGroup(TypeEnum.SHIPYARD, _hullsText);
_accordian.addSubItemToGroup(TypeEnum.SHIPYARD, FilterEnum.SHIP_HULLS,
presenter.getFilterNameByKey(FilterEnum.SHIP_HULLS), 0);
_accordian.addSubItemToGroup(TypeEnum.SHIPYARD, FilterEnum.SHIP_HULLS_SPECIAL,
presenter.getFilterNameByKey(FilterEnum.SHIP_HULLS_SPECIAL), 0);
_accordian.addSubItemToGroup(TypeEnum.SHIPYARD, FilterEnum.INTEGRITY_FIELD,
presenter.getFilterNameByKey(FilterEnum.INTEGRITY_FIELD), 0);
_accordian.addSubItemToGroup(TypeEnum.SHIPYARD, "Blueprint", _blueprints, 0);
_accordian.addGroup(TypeEnum.ADVANCED_TECH, _techText);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH,
FilterEnum.BASE_WEAPONS, presenter.getFilterNameByKey(FilterEnum.BASE_WEAPONS),
0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH, FilterEnum.SHIP_TECH,
presenter.getFilterNameByKey(FilterEnum.SHIP_TECH), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH,
FilterEnum.WEAPONS_TECH, presenter.getFilterNameByKey(FilterEnum.WEAPONS_TECH),
0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH, FilterEnum.BEAM_TECH,
presenter.getFilterNameByKey(FilterEnum.BEAM_TECH), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH,
FilterEnum.PROJECTILE_TECH,
presenter.getFilterNameByKey(FilterEnum.PROJECTILE_TECH), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH, FilterEnum.GUIDED_TECH,
presenter.getFilterNameByKey(FilterEnum.GUIDED_TECH), 0);
//PR: Disabling Secondary Tech for the release
//_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH,
FilterEnum.SECONDARY_TECH,
presenter.getFilterNameByKey(FilterEnum.SECONDARY_TECH), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH,
FilterEnum.DRONE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.DRONE_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH, FilterEnum.LEGACY_TECH,
presenter.getFilterNameByKey(FilterEnum.LEGACY_TECH), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH,
FilterEnum.LEGACY_BASE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_BASE_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.ADVANCED_TECH, "Blueprint", _blueprints, 0);
_accordian.addGroup(TypeEnum.WEAPONS_FACILITY,
```

```
_weaponsText);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.PROJECTILE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.PROJECTILE_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.BEAM_WEAPONS, presenter.getFilterNameByKey(FilterEnum.BEAM_WEAPONS),
0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.GUIDED_WEAPON,
presenter.getFilterNameByKey(FilterEnum.GUIDED_WEAPON), 0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.ARC_WEAPONS, presenter.getFilterNameByKey(FilterEnum.ARC_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.SPINAL_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.SPINAL_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.BASE_WEAPONS, presenter.getFilterNameByKey(FilterEnum.BASE_WEAPONS),
0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.LEGACY_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY,
FilterEnum.LEGACY_BASE_WEAPONS,
presenter.getFilterNameByKey(FilterEnum.LEGACY_BASE_WEAPONS), 0);
_accordian.addSubItemToGroup(TypeEnum.WEAPONS_FACILITY, "Blueprint", _blueprints, 0);

if (!_groupID)
{
if (LAST_VIEWED.hasOwnProperty(_state))
{
_groupID = LAST_VIEWED[_state].groupID;
_subItemID = LAST_VIEWED[_state].subItemID;
} else
_groupID = TypeEnum.DEFENSE_DESIGN;
}
if (!_subItemID)
onAccordianSelected(_groupID, null, null);
else
{
_accordian.setSelected(_groupID, _subItemID);
onAccordianSelected(_groupID, _subItemID, null);
}
break;

case BUILD:
default:
_bg.addTitle(_construction, 300);
_accordian.addGroup(StarbaseCategoryEnum.STARBASE_STRUCTURE,
_starbaseStructureButton);
_accordian.addGroup(StarbaseCategoryEnum.INFRASTRUCTURE, _infrastructureButton);
_accordian.addGroup(StarbaseCategoryEnum.RESEARCH,
```

```
_researchButton);
_accordian.addGroup(StarbaseCategoryEnum.DEFENSE, _defenseButton);
_accordian.addGroup(StarbaseCategoryEnum.FLEETS, _fleetsButton);

//set default
if (!_groupID)
{
if (LAST_VIEWED.hasOwnProperty(_state))
{
_groupID = LAST_VIEWED[_state].groupID;
_subItemID = LAST_VIEWED[_state].subItemID;
} else
_groupID = StarbaseCategoryEnum.INFRASTRUCTURE;
}
_accordian.setSelected(_groupID, null);
onAccordianSelected(_groupID, null, null);
break;
}

if (_specialButton)
{
addListener(_specialButton, MouseEvent.CLICK, onSpecialButtonClicked);
addChild(_specialButton);
}
}

private function onHighestSelected( e:MouseEvent ):void
{
if(_highestButton.selected)
{
_advancedButton.selected = false;
_commonButton.selected = false;
_uncommonButton.selected = false;
_rareButton.selected = false;
_epicButton.selected = false;
_legendaryButton.selected = false;
}
onAccordianSelected(_groupID, _subItemID, null);
}
private function onAdvancedOnlySelected( e:MouseEvent ):void
{
if(_advancedButton.selected)
{
_commonButton.selected = false;
_uncommonButton.selected = false;
_rareButton.selected = false;
_epicButton.selected = false;
_legendaryButton.selected = false;
_highestButton.selected = false;
}
onAccordianSelected(_groupID,
```

```
_subItemID, null);
}
private function onCommonOnlySelected( e:MouseEvent ):void
{
if(_commonButton.selected)
{
_advancedButton.selected = false;
_uncommonButton.selected = false;
_rareButton.selected = false;
_epicButton.selected = false;
_legendaryButton.selected = false;
_highestButton.selected = false;
}
onAccordianSelected(_groupID, _subItemID, null);
}
private function onUncommonOnlySelected( e:MouseEvent ):void
{
if(_uncommonButton.selected)
{
_advancedButton.selected = false;
_commonButton.selected = false;
_rareButton.selected = false;
_epicButton.selected = false;
_legendaryButton.selected = false;
_highestButton.selected = false;
}
onAccordianSelected(_groupID, _subItemID, null);
}
private function onRareOnlySelected( e:MouseEvent ):void
{
if(_rareButton.selected)
{
_advancedButton.selected = false;
_commonButton.selected = false;
_uncommonButton.selected = false;
_epicButton.selected = false;
_legendaryButton.selected = false;
_highestButton.selected = false;
}
onAccordianSelected(_groupID, _subItemID, null);
}
private function onEpicOnlySelected( e:MouseEvent ):void
{
if(_epicButton.selected)
{
_advancedButton.selected = false;
_commonButton.selected = false;
_uncommonButton.selected = false;
_rareButton.selected = false;
_legendaryButton.selected = false;
_highestButton.selected
```

```
= false;
}
onAccordianSelected(_groupID, _subItemID, null);
}
private function onLegendaryOnlySelected( e:MouseEvent ):void
{
if(_legendaryButton.selected)
{
_advancedButton.selected = false;
_commonButton.selected = false;
_uncommonButton.selected = false;
_rareButton.selected = false;
_epicButton.selected = false;
_highestButton.selected = false;
}
onAccordianSelected(_groupID, _subItemID, null);
}
private function onSpecialButtonClicked( e:MouseEvent ):void { _list.onSpecialButtonClicked(e);
}
private function onTransactionChanged( transaction:TransactionVO ):void {
onAccordianSelected(_groupID, _subItemID, null); }

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

public function get state():int { return _state; }

[Inject]
public function set presenter( v:IConstructionPresenter ):void { _presenter = v; }
public function get presenter():IConstructionPresenter { return
IConstructionPresenter(_presenter); }

[Inject]
public function set tooltips( v:Tooltips ):void { _tooltips = v; }

override public function destroy():void
{
presenter.removeOnTransactionRemovedListener(onTransactionChanged);
super.destroy();

ObjectPool.give(_accordian);
_accordian = null;
ObjectPool.give(_bg);
_bg = null;
_closeButton = UIFactory.destroyButton(_closeButton);

_advancedButton = UIFactory.destroyButton(_advancedButton);
_commonButton = UIFactory.destroyButton(_commonButton);
_uncommonButton = UIFactory.destroyButton(_uncommonButton);
_rareButton = UIFactory.destroyButton(_rareButton);
_epicButton
```

```
= UIFactory.destroyButton(_epicButton);
_legendaryButton = UIFactory.destroyButton(_legendaryButton);
_highestButton = UIFactory.destroyButton(_highestButton);
ObjectPool.give(_list);
_list = null;
_groupID = _subItemID = null;
_specialButton = UIFactory.destroyButton(_specialButton);
_tooltips.removeTooltip(null, this);
_tooltips = null;
}
}
}

------------------------------------
File 802: igw\com\ui\modal\credits\CreditsView.as
package com.ui.modal.credits
{
import com.presenter.shared.IUIPresenter;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.label.Label;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import org.greensock.TweenLite;
import org.greensock.easing.Linear;
import org.shared.ObjectPool;

public class CreditsView extends View
{
private var _bg:DefaultWindowBG;

private var _creditsTitleText:Label;
private var _creditsBodyText:Label;

private var _textHolder:Sprite;

private var _scrollRect:Rectangle;

private var _credits:Array;

private var currentCredit:Object;

private var _creditsText:String = 'CodeString.SettingsView.Credits'; //CREDITS

[PostConstruct]
override
```

```
public function init():void
{
super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(722, 475);
_bg.addTitle(_creditsText, 239);
_bg.x -= 21;
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_textHolder = new Sprite();
_textHolder.x = 1;
_textHolder.y = 43;
_textHolder.graphics.beginFill(0xf0f0f0, 0.0);
_textHolder.graphics.drawRect(0, 0, 696, 472);
_textHolder.graphics.endFill();

_scrollRect = new Rectangle(0, 0, 696, 472);
_scrollRect.y = 0;
_textHolder.scrollRect = _scrollRect;

_creditsTitleText = new Label(50, 0xf0f0f0, 722, 100, false);
_creditsTitleText.align = TextFormatAlign.CENTER;

_creditsBodyText = new Label(36, 0xf0f0f0, 722, 475, false);
_creditsBodyText.constrictTextToSize = false;
_creditsBodyText.multiline = true;
_creditsBodyText.autoSize = TextFieldAutoSize.CENTER;
_creditsBodyText.align = TextFormatAlign.CENTER;

_textHolder.addChild(_creditsTitleText);
_textHolder.addChild(_creditsBodyText);

addChild(_bg);
addChild(_textHolder);

addEffects();
effectsIN();

presenter.getFromCache('data/Credits.txt', onCreditsLoaded);
}

private function onCreditsLoaded( credits:Object ):void
{
_credits = credits.Credits.slice();
creditsBegin();
}

private function creditsBegin():void
{
if
```

```
(_credits != null && _credits.length > 0)
{
currentCredit = _credits.shift();
var duration:Number = currentCredit.duration;
_creditsTitleText.text = currentCredit.title;
_creditsBodyText.text = currentCredit.body;

switch (currentCredit.type)
{
case 'Scrolling':
_creditsTitleText.y = _textHolder.y + _scrollRect.height;
_creditsBodyText.y = _creditsTitleText.y + _creditsTitleText.textHeight + 50;
_creditsTitleText.alpha = 1;
_creditsBodyText.alpha = 1;
TweenLite.to(_creditsBodyText, duration, {y:(_textHolder.y - (_creditsBodyText.height + 30)),
onUpdate:onScrollingUpdate, onComplete:creditsEnd, ease:Linear.easeNone})
break;
case 'Fade':
_creditsTitleText.alpha = 0;
_creditsBodyText.alpha = 0;
_creditsTitleText.y = (_textHolder.y + (_textHolder.height - _creditsTitleText.textHeight) * 0.5) -
25;
_creditsBodyText.y = _creditsTitleText.y + _creditsTitleText.textHeight + 10;
duration *= 0.5;
TweenLite.to(_creditsTitleText, duration, {alpha:1})
TweenLite.to(_creditsBodyText, duration, {alpha:1, onComplete:onAlphaInComplete})
break;
}
} else
destroy();
}

private function onScrollingUpdate():void
{
if (_creditsTitleText && _creditsBodyText)
_creditsTitleText.y = _creditsBodyText.y - 60;
}

private function onAlphaInComplete():void
{
if (currentCredit)
{
var duration:Number = currentCredit.duration * 0.5;
TweenLite.to(_creditsTitleText, duration, {alpha:0})
TweenLite.to(_creditsBodyText, duration, {alpha:0, onComplete:creditsEnd})
}
}

private function creditsEnd():void
{
if
```

```
(_credits && _credits.length > 0)
{
creditsBegin();
} else
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( value:IUIPresenter ):void { _presenter = value; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
super.destroy();

if (_bg)
ObjectPool.give(_bg);

_bg = null;


if (_creditsTitleText)
_creditsTitleText.destroy();

_creditsTitleText = null;

if (_creditsBodyText)
{
TweenLite.killTweensOf(_creditsBodyText);
_creditsBodyText.destroy();
}

_creditsBodyText = null;

_textHolder = null;
_scrollRect = null;
_credits = null;

currentCredit = null;
}
}
}

-----------------------------------
File 803: igw\com\ui\modal\dock\CrewButton.as
package com.ui.modal.dock
{
import
```

```
com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.misc.ImageComponent;

import flash.display.BitmapData;
import flash.events.MouseEvent;

import org.greensock.TweenLite;
import org.osflash.signals.Signal;

public class CrewButton extends BitmapButton
{
private var _image:ImageComponent;

public var onSelectCrew:Signal;

public function CrewButton()
{
super.init(UIFactory.getBitmapData('BtnShipUpBMD'),
UIFactory.getBitmapData('BtnShipROBMD'), UIFactory.getBitmapData('BtnShipDownBMD'),
UIFactory.getBitmapData('BtnShipDisabledBMD'), UIFactory.
getBitmapData('BtnShipSelectedBMD'));

onSelectCrew = new Signal(CrewButton);

_image = new ImageComponent();
_image.init(_bitmap.width, _bitmap.height);

addChild(_image);

mouseChildren = false;
}

override protected function onMouse( e:MouseEvent ):void
{
super.onMouse(e);
if (mouseEnabled)
{
switch (e.type)
{
case MouseEvent.CLICK:
onSelectCrew.dispatch(this);
break;
}

}
}

public function onImageLoaded( asset:BitmapData ):void
{
if
```

```
(_image)
{
_image.onImageLoaded(asset);
_image.smoothing = true;
_image.x = _bitmap.x + (_bitmap.width - _image.width) * 0.5;
_image.y = _bitmap.y + (_bitmap.height - _image.height) * 0.5;
}
}

public function clearImageBitmap():void
{
_image.clearBitmap();
}

override public function destroy():void
{
onSelectCrew.removeAll();
onSelectCrew = null;

_image.destroy();
_image = null;

super.destroy();
}

}
}
```

File 804: igw\com\ui\modal\dock\DockView.as

```
package com.ui.modal.dock
{
import com.controller.transaction.requirements.PurchaseVO;
import com.controller.transaction.requirements.RequirementVO;
import com.enum.CurrencyEnum;
import com.enum.FleetStateEnum;
import com.enum.SlotComponentEnum;
import com.enum.ToastEnum;
import com.enum.server.PurchaseTypeEnum;
import com.enum.server.StarbaseTransactionStateEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.model.transaction.TransactionVO;
import com.presenter.starbase.IFleetPresenter;
import com.ui.UIFactory;
import com.ui.core.ButtonPrototype;
import
```

```
com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ActionComponent;
import com.ui.core.component.misc.ActionInProgressComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.hud.shared.command.ResourceComponent;
import com.ui.modal.information.ResourceModalView;
import com.ui.modal.shipyard.ShipyardView;
import com.ui.modal.store.StoreView;
import com.util.CommonFunctionUtil;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.filters.ColorMatrixFilter;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.Timer;

import org.adobe.utils.StringUtil;
import org.shared.ObjectPool;

public class DockView extends View
{
private var DEFAULT_STATE:int = 0;
private var REPAIR_STATE:int = 1;
private var NEEDS_REPAIR_STATE:int = 2;

private var _bg:DefaultWindowBG;

private var _fleetSelection:FleetSelection;

private var _fleetInfoBG:Sprite;
private var _fleetStatsBG:Sprite;

private var _nameBG:ScaleBitmap;
private var _ratingBG:ScaleBitmap;

private var _powerSpent:ProgressBar;
private var _fleetDpsBar:ProgressBar;
private var _fleetHealthBar:ProgressBar;
private var _fleetCargoBar:ProgressBar;
private var _fleetMapSpeedBar:ProgressBar;

private var _repairComponent:ActionComponent;
private
```

```
var _repairInProgressComponent:ActionInProgressComponent;
private var _repairCostComponent:ResourceComponent;

private var _changeFleetNameBtn:BitmapButton;
private var _docksBtnTwo:BitmapButton;
private var _docksBtnOne:BitmapButton;
private var _shipActionBtn:BitmapButton;
private var _shipRefitBtn:BitmapButton;

private var _powerUsedText:Label;
private var _fleetName:Label;
private var _statusText:Label;
private var _statusInfoText:Label;
private var _subTitle:Label;
private var _fleetNameLabel:Label;
private var _fleetRatingLabel:Label;
private var _fleetPowerLabel:Label;
private var _fleetRatingText:Label;
private var _fleetDpsLabel:Label;
private var _fleetHealthLabel:Label;
private var _fleetCargoLabel:Label;
private var _fleetMapSpeedLabel:Label;
private var _fleetDpsValue:Label;
private var _fleetHealthValue:Label;
private var _fleetCargoValue:Label;
private var _fleetMapSpeedValue:Label;
private var _fleetDamageTitle:Label;
private var _fleetDamage:Label;

private var _fleets:Vector.<FleetVO>;
private var _shipButtons:Vector.<ShipButton>;

private var _selectedShipButton:ShipButton;

private var _redFilter:ColorMatrixFilter;
private var _greenFilter:ColorMatrixFilter;
private var _currentFilter:ColorMatrixFilter;

private var _currentState:int;
private var _currentRepairTime:int;

private var _selectedFleet:FleetVO;

private var _timer:Timer;
private var _repairTransaction:TransactionVO;

private var _currentRequirements:RequirementVO;

protected var _speedUpBtnText:String = 'CodeString.Shared.SpeedUp';
protected var _cancelBtnText:String = 'CodeString.Shared.CancelBtn';

private
```

```
var _titleText:String = 'CodeString.Docks.Title'; //DOCKS
private var _level:String = 'CodeString.Shared.Level'; //Level [[Number.Level]]
private var _mapSpeedText:String = 'CodeString.Shared.MapSpeed'; // Map Speed
[[Number.WorldMapSpeed]]
private var _cargoText:String = 'CodeString.Shared.Cargo'; //Cargo [[Number.MaxCargoCount]]
private var _cancelRepair:String = 'CodeString.Shared.CancelBtn'; //Cancel
private var _repair:String = 'CodeString.Docks.RepairBtn'; //Repair
private var _repairNow:String = 'CodeString.Docks.RepairNowBtn'; //Repair Now
private var _launch:String = 'CodeString.Docks.LaunchFleetBtn'; //Launch Fleet
private var _defend:String = 'CodeString.Docks.DefendBaseBtn'; //Defend Base
private var _offline:String = 'CodeString.Docks.OfflineBtn'; //Offline
private var _noFlagship:String = 'CodeString.Docks.NoFlagShipBtn'; //No Flagship
private var _needToRepair:String = 'CodeString.Docks.NeedToRepairBtn'; //Need To Repair
private var _recall:String = 'CodeString.Docks.RecallFleetBtn'; //Recall Fleet
private var _gotoFleet:String = 'CodeString.Docks.GotoFleetBtn'; //Goto Fleet
private var _outOfString:String = 'CodeString.Shared.OutOf';
//[[Number.MinValue]]/[[Number.MaxValue]]
private var _shipyardBusyAlertTitle:String = 'CodeString.Alert.ShipyardBusy.Title'; //Shipyard
Busy
private var _shipyardBusyAlertBody:String = 'CodeString.Alert.ShipyardBusy.Body'; //Your
Shipyard is currently Busy. Would you like to speed up it's current transaction?
private var _changeFleetNameAlertTitle:String = 'CodeString.Alert.ChangeFleetName.Title';
//Change Fleet Name
private var _changeFleetNameAlertBody:String = 'CodeString.Alert.ChangeFleetName.Body';
//Please enter a name for your fleet
private var _changeFleetNameAlertCancel:String = 'CodeString.Shared.CancelBtn'; //Cancel
private var _changeFleetNameAlertAccept:String = 'CodeString.Shared.OkBtn'; //Ok
private var _noFleetAlertTitle:String = 'CodeString.Alert.NoShips.Title'; //No Unassigned Ships
private var _noFleetAlertBody:String = 'CodeString.Alert.NoShips.Body'; //Sorry but you have no
unassigned ships to assign to fleets, but you can build more at your shipyard!
private var _noFleetAlertClose:String = 'CodeString.Shared.CloseBtn'; //Close
private var _noFleetAlertOpenShipyard:String = 'CodeString.Alert.NoShips.OpenShipyardBtn';
//Open Shipyard
private var _speedUpText:String = 'CodeString.Shared.SpeedUp'; //Speed Up
private var _readyStatusText:String = 'CodeString.Docks.Status.Ready'; //READY!
private var _offlineStatusText:String = 'CodeString.Docks.Status.Offline'; //OFFLINE!
private var _BattleStatusText:String = 'CodeString.Docks.Status.InCombat'; //IN COMBAT!
private var _statusTitle:String = 'CodeString.Docks.Status'; //STATUS
private var _statusStandingBy:String = 'CodeString.Docks.Status.StandingBy'; //STANDING BY!
private var _damageTitleText:String = 'CodeString.Shared.DamageTitle'; //Damage
private var _healthTitleText:String = 'CodeString.Shared.HealthTitle'; //Health
private var _cargoTitleText:String = 'CodeString.Shared.CargoTitle'; //Cargo
private var _loadSpeedTitleText:String = 'CodeString.Shared.LoadSpeedTitle'; //Load Speed
private var _mapSpeedTitleText:String = 'CodeString.Shared.MapSpeedTitle'; //Map Speed
private var _percentText:String = 'CodeString.Shared.Percent'; // [[Number.PercentValue]]%
private var _auPerHr:String = 'CodeString.Shared.AuPerHr'; //[[Number.ValuePerHr]] au/h
private var _perSecond:String = 'CodeString.Shared.PerSecond';
//[[Number.ValuePerSecond]]/s
private var _alertBodyBuyResources:String = 'CodeString.Alert.BuyResources.Body';
private var _alertHeadingBuyResources:String = 'CodeString.Alert.BuyResources.Title';

private
```

```
var _remove:String = 'CodeString.Docks.ShipSelection.Remove'; //Remove
private var _add:String = 'CodeString.Docks.ShipSelection.Add'; //Select
private var _refitShipBtnText:String = 'CodeString.Shipyard.RefitShipBtn'; //REFIT

private var _subtitleText:String = 'CodeString.Dock.Subtitle'; //UPGRADE DOCK TO UNLOCK
MORE FLEETS
private var _fleetNameText:String = 'CodeString.Dock.FleetName'; //FLEET NAME:
private var _FleetRatingLabelText:String = 'CodeString.Dock.FleetRating'; //FLEET RATING:
private var _FleetPowerText:String = 'CodeString.Dock.FleetPower'; //FLEET POWER:
private var _fleetInfoText:String = 'CodeString.Dock.FleetInfo'; //FLEET INFO
private var _fleetStatsText:String = 'CodeString.Dock.FleetStats'; //FLEET STATS
private var _fleetHealthText:String = 'CodeString.Dock.FleetHealth'; //FLEET HEALTH

[Inject]
public var tooltip:Tooltips;

[PostConstruct]
override public function init():void
{
super.init();

_greenFilter = CommonFunctionUtil.getColorMatrixFilter(0x48b53c);
_redFilter = CommonFunctionUtil.getColorMatrixFilter(0xf81919);

_timer = new Timer(1000)
_timer.addEventListener(TimerEvent.TIMER, onRepairTick, false, 0, true);

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(959, 535);
_bg.addTitle(_titleText, 470);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_subTitle = new Label(26, 0xffd785, 352, 45);
_subTitle.align = TextFormatAlign.RIGHT;
_subTitle.x = 140;
_subTitle.y = 7;
_subTitle.text = _subtitleText;
_subTitle.visible = presenter.dockLevel < 10;

_fleetSelection = new FleetSelection();
_fleetSelection.x = 30;
_fleetSelection.y = 53;
_fleetSelection.loadShipImage = presenter.loadIconFromEntityData;
_fleetSelection.onFleetSelected.add(onFleetSelected);

_fleetInfoBG = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_INNER,
PanelEnum.HEADER_NOTCHED, 563, 146, 32, 384, 133, _fleetInfoText, LabelEnum.H1);
_fleetInfoBG.x = 398;
_fleetInfoBG.y = 144;

_fleetStatsBG
```

```
= UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED,
PanelEnum.HEADER_NOTCHED, 563, 205, 32, 384, 319, _fleetStatsText, LabelEnum.H1);
_fleetStatsBG.x = 398;
_fleetStatsBG.y = 330;

_docksBtnOne = UIFactory.getButton(ButtonEnum.BLUE_A, 239, 42, 478, 582, _defend,
LabelEnum.H1);
addListener(_docksBtnOne, MouseEvent.CLICK, onDockBtnOneClick);

_docksBtnTwo = UIFactory.getButton(ButtonEnum.BLUE_A, 239, 42, 728, 582, _launch,
LabelEnum.H1);
addListener(_docksBtnTwo, MouseEvent.CLICK, onDockBtnTwoClick);

_repairInProgressComponent = new ActionInProgressComponent(new
ButtonPrototype(_cancelRepair, onCancelRepairClick), new ButtonPrototype(_speedUpText,
onSpeedUpClick));
_repairInProgressComponent.visible = false;
_repairInProgressComponent.x = 681;
_repairInProgressComponent.y = 257;

_repairCostComponent = ObjectPool.get(ResourceComponent);
_repairCostComponent.init(false, false, 35);
_repairCostComponent.addMoreListener(onPurchaseMoreResources);
_repairCostComponent.visible = false;
_repairCostComponent.x = 405;
_repairCostComponent.y = 251;

_repairComponent = new ActionComponent(new ButtonPrototype(_repair, onRepairClick), new
ButtonPrototype(_repairNow, onRepairNowClick), new
ButtonPrototype(_alertHeadingBuyResources, onClickCannotAffordResourceDialog),
new ButtonPrototype(_repairNow, popPaywall));
_repairComponent.visible = false;
_repairComponent.x = 695;
_repairComponent.y = 257;

_nameBG = UIFactory.getScaleBitmap(PanelEnum.STATBAR_CONTAINER);
_nameBG.width = 263;
_nameBG.height = 31;
_nameBG.x = 492;
_nameBG.y = 185;

_fleetName = new Label(20, 0xf0f0f0, 228, 30, false);
_fleetName.align = TextFormatAlign.LEFT;
_fleetName.x = _nameBG.x + 6;
_fleetName.y = _nameBG.y + 4;

_changeFleetNameBtn = UIFactory.getButton(ButtonEnum.EDIT);
_changeFleetNameBtn.addEventListener(MouseEvent.CLICK, onChangeFleetName, false, 0,
true);
_changeFleetNameBtn.x = _nameBG.x + _nameBG.width - (_changeFleetNameBtn.width + 4);
_changeFleetNameBtn.y
```

```
= _nameBG.y + (_nameBG.height - _changeFleetNameBtn.height) * 0.5;

_ratingBG = UIFactory.getScaleBitmap(PanelEnum.STATBAR_CONTAINER);
_ratingBG.width = 54;
_ratingBG.height = 31;
_ratingBG.x = 895;
_ratingBG.y = 185;

_fleetRatingText = new Label(20, 0xf0f0f0, 54, 31, false);
_fleetRatingText.align = TextFormatAlign.CENTER;
_fleetRatingText.constrictTextToSize = false;
_fleetRatingText.x = _ratingBG.x;
_fleetRatingText.y = _ratingBG.y + 4;

_powerSpent = UIFactory.getProgressBar(UIFactory.getPanel(PanelEnum.STATBAR_GREY,
448, 20), UIFactory.getPanel(PanelEnum.STATBAR_CONTAINER, 456, 26), 0, 1, 0, 492, 223);
_powerSpent.overlay.y += 1;

_powerUsedText = new Label(20, 0xf0f0f0, _powerSpent.width, _powerSpent.height);
_powerUsedText.letterSpacing = 1.5;
_powerUsedText.constrictTextToSize = false;
_powerUsedText.align = TextFormatAlign.CENTER;
_powerUsedText.x = _powerSpent.x;
_powerUsedText.y = _powerSpent.y + 1;

_fleetNameLabel = new Label(18, 0xf0f0f0, 93, 30);
_fleetNameLabel.align = TextFormatAlign.CENTER;
_fleetNameLabel.x = _fleetInfoBG.x + 1;
_fleetNameLabel.y = _fleetInfoBG.y + 45;
_fleetNameLabel.text = _fleetNameText;

_fleetRatingLabel = new Label(18, 0xf0f0f0, 93, 30);
_fleetRatingLabel.align = TextFormatAlign.CENTER;
_fleetRatingLabel.x = _fleetInfoBG.x + 402;
_fleetRatingLabel.y = _fleetInfoBG.y + 45;
_fleetRatingLabel.text = _FleetRatingLabelText;

_fleetPowerLabel = new Label(18, 0xf0f0f0, 93, 30);
_fleetPowerLabel.align = TextFormatAlign.CENTER;
_fleetPowerLabel.x = _fleetInfoBG.x + 1;
_fleetPowerLabel.y = _fleetInfoBG.y + 80;
_fleetPowerLabel.text = _FleetPowerText;

_shipActionBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 170, 42, 35, 520, _add,
LabelEnum.H1);
addListener(_shipActionBtn, MouseEvent.CLICK, onActionBtnClicked);

_shipRefitBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 170, 42, 218, 520,
_refitShipBtnText, LabelEnum.H1);
addListener(_shipRefitBtn, MouseEvent.CLICK, onRefitShip);

_statusText
```

```
= new Label(16, 0xFFFFFF, 140, 20, true, 1);
_statusText.constrictTextToSize = false;
_statusText.autoSize = TextFieldAutoSize.CENTER;
_statusText.x = 99;
_statusText.y = 144;
_statusText.text = _statusTitle;

_statusInfoText = new Label(14, 0x95c5f4, 140, 20, true, 1);
_statusInfoText.constrictTextToSize = false;
_statusInfoText.autoSize = TextFieldAutoSize.CENTER;
_statusInfoText.x = 99;
_statusInfoText.y = 162;
_statusInfoText.text = _readyStatusText;

_fleetDamageTitle = new Label(16, 0xFFFFFF, 140, 20, true, 1);
_fleetDamageTitle.constrictTextToSize = false;
_fleetDamageTitle.autoSize = TextFieldAutoSize.CENTER;
_fleetDamageTitle.x = 317;
_fleetDamageTitle.y = 144;
_fleetDamageTitle.text = _fleetHealthText;

_fleetDamage = new Label(14, 0x95c5f4, 140, 20, true, 1);
_fleetDamage.constrictTextToSize = false;
_fleetDamage.autoSize = TextFieldAutoSize.CENTER;
_fleetDamage.x = 317;
_fleetDamage.y = 162;

addChild(_bg);
addChild(_subTitle);
addChild(_fleetSelection);
addChild(_fleetInfoBG);
addChild(_fleetStatsBG);
addChild(_nameBG);
addChild(_changeFleetNameBtn);
addChild(_fleetName);
addChild(_ratingBG);
addChild(_fleetRatingText);
addChild(_powerSpent);
addChild(_powerUsedText);
addChild(_fleetNameLabel);
addChild(_fleetRatingLabel);
addChild(_fleetPowerLabel);
addChild(_statusInfoText);
addChild(_statusText);
addChild(_fleetDamageTitle);
addChild(_fleetDamage);
addChild(_repairInProgressComponent);
addChild(_repairCostComponent);
addChild(_repairComponent);
addChild(_docksBtnOne);
addChild(_docksBtnTwo);
```

```
addChild(_shipActionBtn);
addChild(_shipRefitBtn);

createStatBar("_fleetDpsBar", "_fleetDpsLabel", "_fleetDpsValue", 'shipDps', 409, 366);
createStatBar("_fleetHealthBar", "_fleetHealthLabel", "_fleetHealthValue", 'health', 409, 414);
createStatBar("_fleetCargoBar", "_fleetCargoLabel", "_fleetCargoValue", 'cargo', 409, 461);
createStatBar("_fleetMapSpeedBar", "_fleetMapSpeedLabel", "_fleetMapSpeedValue",
'mapSpeed', 409, 508);

_fleets = presenter.fleets;
_fleetSelection.addFleets(_fleets);

_shipButtons = new Vector.<ShipButton>;
var currentShipButton:ShipButton;
var xPos:Number;
var yPos:Number;
for (var i:uint = 0; i < 6; ++i)
{
currentShipButton = new ShipButton();
currentShipButton.onLoadShipImage.add(presenter.loadIconFromEntityData);
currentShipButton.index = i;
currentShipButton.onSelectShip.add(onSelectShip);
tooltip.addTooltip(currentShipButton, this, currentShipButton.getTooltip, '', 250, 180, 14);

switch (i)
{
case 0:
xPos = 146;
yPos = 153;
break;
case 1:
xPos = 37;
yPos = 211;
break;
case 2:
xPos = 254;
yPos = 211;
break;
case 3:
xPos = 37;
yPos = 335;
break;
case 4:
xPos = 254;
yPos = 335;
break;
case 5:
xPos = 146;
yPos = 391;
break;
```

```
}

currentShipButton.x = xPos;
currentShipButton.y = yPos;

addChild(currentShipButton);
_shipButtons.push(currentShipButton);
}

_fleetSelection.setSelected(presenter.selectedFleetID);
presenter.addTransactionListener(transactionUpdate);
presenter.addListenerOnFleetUpdated(onFleetsUpdated);
transactionUpdate();
addEffects();
effectsIN();
}


//==========================================================================
//**************************************************************************************************
// FLEET CONTROL
//**************************************************************************************************

//==========================================================================

private function gotoFleet():void
{
presenter.gotoFleet(_selectedFleet);
destroy();
}

private function recallFleet():void
{
presenter.recallFleet(_selectedFleet.id);
_selectedFleet.state = FleetStateEnum.DOCKING;
update();
}

private function defendBase():void
{
_selectedFleet.state = FleetStateEnum.DEFENDING;
update();
}

private function stopDependingBase():void
{
_selectedFleet.state = FleetStateEnum.DOCKED;
update();
}

private
```

```
function launchFleet():void
{
var fleetsToLaunch:Array = _fleetSelection.queuedFleets;

var index:int = fleetsToLaunch.indexOf(_selectedFleet.id);
if (index == -1)
fleetsToLaunch.unshift(_selectedFleet);

if (fleetsToLaunch.length > 0)
{
presenter.launchFleet(fleetsToLaunch);
destroy();
}
}

private function onChangeFleetName( e:MouseEvent ):void
{
showInputAlert(_changeFleetNameAlertTitle, _changeFleetNameAlertBody,
_changeFleetNameAlertCancel, null, null, _changeFleetNameAlertAccept, onChangedName,
null, true, 20, _fleetName.text);
}

private function onChangedName( newName:String ):void
{
setFleetName(newName);
if (presenter)
presenter.changeFleetName(_selectedFleet, newName);
}

private function setFleetName( v:String ):void
{
if (_fleetName)
_fleetName.text = v;
}


//==========================================================================
//**************************************************************************************************
// REPAIRING
//**************************************************************************************************

//==========================================================================

private function updateNeedsRepairState( show:Boolean ):void
{
_repairComponent.visible = show;
_repairCostComponent.visible = show;

if (show)
{
_selectedFleet.updateFleetStats();
```

```
_repairComponent.timeCost = _selectedFleet.repairTime;
}

}

private function updateRepairState( show:Boolean ):void
{
if (show)
onRepairTick(null);

_repairInProgressComponent.visible = show;
}

private function onRepairTick( e:TimerEvent ):void
{
if (_repairTransaction)
{
_repairInProgressComponent.timeRemaining = _repairTransaction.timeRemainingMS;

if (_repairTransaction.timeRemainingMS <= 0)
_timer.reset();
else
presenter.updateRepair();

if (_repairTransaction.id == _selectedFleet.id)
{
updateShips(_repairTransaction && _repairTransaction.id == _selectedFleet.id);
_fleetDamage.text = Math.round(_selectedFleet.currentHealth * 100) + '%';
}

onFleetUpdated(presenter.getFleet(_repairTransaction.id));
}
}

private function onRepairClick( e:MouseEvent ):void
{
presenter.repairFleet(_selectedFleet, PurchaseTypeEnum.NORMAL);
}

private function onRepairNowClick( e:MouseEvent ):void
{
presenter.repairFleet(_selectedFleet, PurchaseTypeEnum.INSTANT);
}

private function onRepairWithResourcePurchase( e:MouseEvent ):void
{
repairWithResourcePurchase();
}

private
```

```
function repairWithResourcePurchase():void
{
presenter.repairFleet(_selectedFleet, PurchaseTypeEnum.GET_RESOURCES);
}

private function onCancelRepairClick( e:MouseEvent ):void
{
presenter.cancelTransaction(_repairTransaction);
}


//==========================================================================================
//******************************************************************************************
// STATE
//******************************************************************************************

//==========================================================================================

private function onFleetSelected( fleetBtn:FleetButton ):void
{
if (fleetBtn)
{
if (_selectedShipButton)
{
_selectedShipButton.selectable = true;
_selectedShipButton.selected = false;
_selectedShipButton = null;
}

presenter.selectedFleetID = fleetBtn.id;
_selectedFleet = fleetBtn.fleet;
setFleetName(_selectedFleet.name);
updateState();
}
}

private function updateState( state:int = 0 ):void
{
_repairTransaction = presenter.dockTransaction;
if (_repairTransaction && (_repairTransaction.timeMS > 0 || _repairTransaction.state ==
StarbaseTransactionStateEnum.PENDING))
state = REPAIR_STATE;
else if (_selectedFleet.needsRepair && _selectedFleet.sector == '')
state = NEEDS_REPAIR_STATE;
else
state = DEFAULT_STATE;

if (state != _currentState)
cleanUpPreviousState();
_currentState = state;
update();
```

```
switch (state)
{
case NEEDS_REPAIR_STATE:
updateNeedsRepairState(true);
break;
case REPAIR_STATE:
updateRepairState(true);
break;
}
}

private function cleanUpPreviousState():void
{
switch (_currentState)
{
case NEEDS_REPAIR_STATE:
updateNeedsRepairState(false);
break;
case REPAIR_STATE:
updateRepairState(false);
break;
}
}

private function transactionUpdate( data:* = null ):void
{
var i:uint;
var len:uint;
updateState();
if (_repairTransaction)
{
if (!_timer.running)
{
presenter.updateRepair();
_fleets = presenter.fleets;
len = _fleets.length;
for (i = 0; i < len; ++i)
onFleetUpdated(_fleets[i]);

_timer.start();
}
} else if (_timer.running)
{
_timer.reset();
presenter.updateRepair();
_fleets = presenter.fleets;
len = _fleets.length;
for (i = 0; i < len; ++i)
onFleetUpdated(_fleets[i]);
}
```

```
}

private function update():void
{
_selectedFleet = presenter.getFleet(_selectedFleet.id);
onFleetUpdated(_selectedFleet);
_selectedFleet.updateFleetStats();

var maxFleetPower:int = presenter.maxFleetPower;
_currentRequirements = presenter.canRepair(_selectedFleet);

var premCost:int = _currentRequirements.purchaseVO.premium;
_repairComponent.instantCost = premCost;
_repairComponent.requirements = _currentRequirements;

_repairCostComponent.updateCost(_selectedFleet.alloyCost,
(_currentRequirements.purchaseVO.alloyAmountShort == 0), CurrencyEnum.ALLOY);
_repairCostComponent.updateCost(_selectedFleet.creditsCost,
(_currentRequirements.purchaseVO.creditsAmountShort == 0), CurrencyEnum.CREDIT);
_repairCostComponent.updateCost(_selectedFleet.energyCost,
(_currentRequirements.purchaseVO.energyAmountShort == 0), CurrencyEnum.ENERGY);
_repairCostComponent.updateCost(_selectedFleet.syntheticCost,
(_currentRequirements.purchaseVO.syntheticAmountShort == 0), CurrencyEnum.SYNTHETIC);

var isRepairing:Boolean;
if (_repairTransaction && _repairTransaction.id == _selectedFleet.id)
isRepairing = true;

updateShips(isRepairing);

if (_selectedFleet.numOfShips == 0)
onSelectShip(_shipButtons[0]);

_fleetRatingText.text = String(_selectedFleet.level);
_fleetDpsBar.amount = _selectedFleet.damage;
_fleetHealthBar.amount = _selectedFleet.healthAmount;
_fleetCargoBar.amount = _selectedFleet.maxCargo;
_fleetMapSpeedBar.amount = _selectedFleet.sectorSpeed;
_fleetDamage.text = Math.round(_selectedFleet.currentHealth * 100) + '%';

_fleetDpsValue.text = StringUtil.commaFormatNumber(_selectedFleet.damage);
_fleetHealthValue.text =
StringUtil.commaFormatNumber(Math.round(_selectedFleet.healthAmount));
_fleetCargoValue.text = StringUtil.commaFormatNumber(_selectedFleet.maxCargo);
_fleetMapSpeedValue.setTextWithTokens(_auPerHr,
{'[[Number.ValuePerHr]]':_selectedFleet.sectorSpeed});

var massPercent:Number = _selectedFleet.powerUsage / maxFleetPower;

var
```

```
dockBtnOneEnabled:Boolean;
var dockBtnTwoEnabled:Boolean;
if (_selectedFleet.sector != '' || _selectedFleet.state == FleetStateEnum.DEFENDING)
{
_docksBtnTwo.text = _recall;
_docksBtnOne.text = _gotoFleet;
if (!_selectedFleet.inBattle)
_statusInfoText.text = _readyStatusText;
else
_statusInfoText.text = _BattleStatusText;

if (_selectedFleet.state == FleetStateEnum.DEFENDING)
dockBtnOneEnabled = false;
else
dockBtnOneEnabled = true;

if (_selectedFleet.state != FleetStateEnum.DOCKING && _selectedFleet.state !=
FleetStateEnum.FORCED_RECALLING)
dockBtnTwoEnabled = true;
} else if (_selectedFleet.numOfShips > 0 && _selectedFleet.ships[0] == null)
{
_statusInfoText.text = _readyStatusText;
_docksBtnTwo.text = _noFlagship;
_docksBtnOne.text = _noFlagship;
} else if (_selectedFleet.currentHealth == 0 && _selectedFleet.numOfShips > 0)
{
_statusInfoText.text = _readyStatusText;
_docksBtnTwo.text = _needToRepair;
_docksBtnOne.text = _needToRepair;
} else if (_selectedFleet.numOfShips == 0)
{
_statusInfoText.text = _offlineStatusText;
_docksBtnTwo.text = _offline;
_docksBtnOne.text = _offline;
} else
{
_statusInfoText.text = _readyStatusText;
_docksBtnTwo.text = _launch;
_docksBtnOne.text = _defend;
dockBtnOneEnabled = isRepairing ? false : false;
dockBtnTwoEnabled = isRepairing ? false : (massPercent <= 1);
}

_docksBtnOne.enabled = dockBtnOneEnabled;
_docksBtnTwo.enabled = dockBtnTwoEnabled;

_changeFleetNameBtn.visible = (_selectedFleet.sector == '');

if (_powerSpent.amount != massPercent)
{
_powerSpent.amount
```

```
= massPercent;
var newFilter:ColorMatrixFilter = getPowerBarColor(massPercent);
if (_currentFilter != newFilter)
{
_currentFilter = newFilter
_powerSpent.overlay.filters = [_currentFilter];
}

_powerUsedText.setTextWithTokens(_outOfString,
{'[[Number.MinValue]]':StringUtil.commaFormatNumber(_selectedFleet.powerUsage),
'[[Number.MaxValue]]':StringUtil.commaFormatNumber(maxFleetPower)});
}
}

private function updateShips( isRepairing:Boolean ):void
{
if (_selectedFleet)
{
var ships:Vector.<ShipVO> = _selectedFleet.ships;
var len:uint = ships.length;
var selectable:Boolean = (_selectedFleet.sector == '' && ((_repairTransaction &&
_repairTransaction.id != _selectedFleet.id) || (_repairTransaction == null)));

if (!selectable && _selectedShipButton)
{
_selectedShipButton.selected = false;
_selectedShipButton = null;
}

var enableShipButtons:Boolean = true;
if (_selectedFleet.numOfShips < 1)
{
if (_selectedFleet.ships[0] == null)
enableShipButtons = false;
}

var shipVO:ShipVO;
var shipBtn:ShipButton;
for (var i:uint = 0; i < len; ++i)
{
shipVO = ships[i];
shipBtn = _shipButtons[i];

shipBtn.setShip(shipVO, null, _selectedFleet);
shipBtn.showRepair(isRepairing);

shipBtn.selectable = selectable;

if (i != 0)
shipBtn.enabled = enableShipButtons;
else
```

```
{
if (_selectedShipButton == null && selectable)
{
shipBtn.selected = true;
onSelectShip(shipBtn);
}
}
}

updateShipActionBtns();
}
}

private function updateShipActionBtns():void
{
var selectable:Boolean;

//For repair time split testing
if (PrototypeModel.instance.getConstantPrototypeValueByName("isNewRepairSystemActive"))
selectable = (_selectedFleet.sector == '' && ((_repairTransaction && _repairTransaction.id !=
_selectedFleet.id) || (_repairTransaction == null)) && _selectedFleet.currentHealth >= 1);
else
selectable = (_selectedFleet.sector == '' && ((_repairTransaction && _repairTransaction.id !=
_selectedFleet.id) || (_repairTransaction == null)));
_shipActionBtn.visible = selectable;
_shipRefitBtn.visible = (selectable && _selectedShipButton && _selectedShipButton.ship &&
_selectedShipButton.ship.currentHealth == 1);
}

private function onFleetUpdated( fleet:FleetVO ):void
{
if (fleet)
_fleetSelection.updateFleet(fleet, (_repairTransaction && _repairTransaction.id == fleet.id));
}

private function onFleetsUpdated( fleet:FleetVO ):void
{
if (fleet == null)
{
_fleets = presenter.fleets;
var len:uint = _fleets.length;
for (var i:uint = 0; i < len; ++i)
{
fleet = _fleets[i];
if (_selectedFleet && fleet.id == _selectedFleet.id)
update();
else
onFleetUpdated(fleet);
}

}
```

```
else
onFleetUpdated(fleet);

}


//=============================================================================================
//*********************************************************************************************
// MISC
//*********************************************************************************************

//=============================================================================================

private function onPurchaseMoreResources( type:String ):void
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
switch (type)
{
case CurrencyEnum.ALLOY:
nStoreView.openToResourcesAndFilter(StoreView.FILTER_ALLOY);
break;
case CurrencyEnum.CREDIT:
nStoreView.openToResourcesAndFilter(StoreView.FILTER_CREDITS);
break;
case CurrencyEnum.ENERGY:
nStoreView.openToResourcesAndFilter(StoreView.FILTER_ENERGY);
break;
case CurrencyEnum.SYNTHETIC:
nStoreView.openToResourcesAndFilter(StoreView.FILTER_SYNTHETIC);
break;
}
}

private function onRefitShip( e:MouseEvent ):void
{
if (_selectedShipButton && _selectedShipButton.ship)
{
var shipyardTransaction:TransactionVO = presenter.shipyardTransaction;
if (_selectedFleet.sector == '' && shipyardTransaction == null)
{
var nShipyardView:ShipyardView = ShipyardView(_viewFactory.createView(ShipyardView));
nShipyardView.refitShip = _selectedShipButton.ship;
_viewFactory.notify(nShipyardView);
} else
{
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_speedUpBtnText, speedUpShipyardTransaction, null, true,
ButtonEnum.GREEN_A));
buttons.push(new ButtonPrototype(_cancelBtnText));
showConfirmation(_shipyardBusyAlertTitle,
```

```
_shipyardBusyAlertBody, buttons);
}
}
}

private function onSpeedUpClick( e:MouseEvent ):void
{
if (_repairTransaction)
openStoreToTransaction(_repairTransaction)
}

private function speedUpShipyardTransaction():void
{
var shipyardTransaction:TransactionVO = presenter.shipyardTransaction;
if (shipyardTransaction)
openStoreToTransaction(shipyardTransaction);
}

private function openStoreToTransaction( transaction:TransactionVO ):void
{
var nStoreView:StoreView = StoreView(_viewFactory.createView(StoreView));
_viewFactory.notify(nStoreView);
nStoreView.setSelectedTransaction(transaction);
destroy();
}

private function onDockBtnOneClick( e:MouseEvent ):void
{
if (_selectedFleet.numOfShips > 0 && _selectedFleet.ships[0] != null &&
_selectedFleet.currentHealth != 0)
{
if (_selectedFleet.sector != '')
gotoFleet();
else if (_selectedFleet.state == FleetStateEnum.DEFENDING)
stopDependingBase();
else
defendBase();
}
}

private function onDockBtnTwoClick( e:MouseEvent ):void
{
if (_selectedFleet.numOfShips > 0 && _selectedFleet.ships[0] != null &&
_selectedFleet.currentHealth != 0)
{
var modules:Dictionary = new Dictionary();
var slots:Array = new Array();
var slot:String;
var shipCount:int = 0;
var hasWeaponCount:int = 0;

if
```

```
(_selectedFleet.sector != '')
recallFleet();
else if (_selectedFleet.state == FleetStateEnum.DEFENDING)
stopDependingBase();
else
{
for each (var ship:ShipVO in _selectedFleet.ships)
{
if (ship)
{
shipCount++;
modules = ship.modules;
slots = ship.slots;
for (var i:int = 0; i < slots.length; i++)
{
slot = slots[i];

if (slot.indexOf(SlotComponentEnum.SLOT_TYPE_WEAPON) != -1)
{
if (modules.hasOwnProperty(slot) && modules[slot] != null)
{
hasWeaponCount++;
break;
}
}
}
}

}

if (hasWeaponCount == shipCount)
launchFleet();
else
showToast(ToastEnum.WRONG, null, 'CodeString.DockView.NoWeaponError');
}




}
}

protected function onClickCannotAffordResourceDialog( e:MouseEvent ):void
{
if (_currentRequirements.purchaseVO.canPurchaseResourcesWithPremium)
{
var purchaseVO:PurchaseVO = _currentRequirements.purchaseVO;
var view:ResourceModalView =
ResourceModalView(_viewFactory.createView(ResourceModalView));
_viewFactory.notify(view);
view.setUp(purchaseVO.creditsAmountShort, purchaseVO.alloyAmountShort,
purchaseVO.energyAmountShort,
```

```
purchaseVO.syntheticAmountShort, 'CodeString.Alert.BuyResources.Title',
'CodeString.Alert.BuyResources.Body',
false, repairWithResourcePurchase, purchaseVO.resourcePremiumCost);
} else
popPaywall();
}

private function getPowerBarColor( v:Number ):ColorMatrixFilter
{
if (v > 1.0)
return _redFilter;
else
return _greenFilter;
}

private function createStatBar( barRef:String, labelRef:String, valueRef:String,
protoName:String, x:int, y:int ):void
{
var statProto:IPrototype = presenter.getStatPrototypeByName(protoName);
var defaultMaxValue:Number =
presenter.getConstantPrototypeValueByName("interfaceCalibrationDefaultStatValue");
var maxValue:Number = statProto.getUnsafeValue("stdMax");
maxValue = (maxValue) ? maxValue : defaultMaxValue;

if (protoName == 'shipDps' || protoName == 'health' || protoName == 'cargo')
maxValue *= 6;

this[labelRef] = new Label(14, 0xf0f0f0, 248, 25, true, 1);
this[labelRef].align = TextFormatAlign.LEFT;
this[labelRef].text = statProto.getValue("lableLocKey");
this[labelRef].y = y;
this[labelRef].x = x;
this[labelRef].constrictTextToSize = false;
addChild(this[labelRef]);

this[barRef] = UIFactory.getProgressBar(UIFactory.getPanel(PanelEnum.STATBAR, 531, 16),
UIFactory.getPanel(PanelEnum.STATBAR_CONTAINER, 539, 24), 0, maxValue, 0, x,
this[labelRef].y + this[labelRef].
textHeight + 4);
this[barRef]
addChild(this[barRef]);

this[valueRef] = new Label(14, 0xf0f0f0, 539, 25, true, 1);
this[valueRef].letterSpacing = 1.5;
this[valueRef].align = TextFormatAlign.CENTER;
this[valueRef].y = this[barRef].y;
this[valueRef].x = this[barRef].x;
this[valueRef].constrictTextToSize = false;
addChild(this[valueRef]);
}

private
```

```
function showShipyardView():void
{
destroy();
showView(ShipyardView);
}

private function onRemovedShipFromFleet( ship:ShipVO ):void
{
presenter.removeShipFromFleet(_selectedFleet, ship.id);

if (_selectedFleet.numOfShips == 0)
onSelectShip(_shipButtons[0]);

_shipActionBtn.text = _add;
}

private function onSelectShip( shipBtn:ShipButton, autoSelect:Boolean = false ):void
{
if (shipBtn)
{
if (_selectedShipButton)
{
_selectedShipButton.selectable = true;
_selectedShipButton.selected = false;
}

_selectedShipButton = shipBtn;
_selectedShipButton.selected = true;
_selectedShipButton.selectable = false;

if (_selectedShipButton.ship)
_shipActionBtn.text = _remove;
else
{
if (autoSelect)
onActionBtnClicked();

_shipActionBtn.text = _add;
}

updateShipActionBtns();
}
}

private function onActionBtnClicked( e:MouseEvent = null ):void
{
if (_selectedShipButton)
{
if (_selectedShipButton.ship)
onRemovedShipFromFleet(_selectedShipButton.ship)
else
```

```
openShipSelect();
}
}

private function openShipSelect():void
{
var unassignedShips:Vector.<ShipVO> = presenter.unassignedShips;
if (unassignedShips.length == 0 || (unassignedShips.length == 1 && !unassignedShips[0].built))
{
var buttons:Vector.<ButtonPrototype> = new Vector.<ButtonPrototype>;
buttons.push(new ButtonPrototype(_noFleetAlertOpenShipyard, showShipyardView, null, true,
ButtonEnum.GREEN_A));
buttons.push(new ButtonPrototype(_noFleetAlertClose));
showConfirmation(_noFleetAlertTitle, _noFleetAlertBody, buttons);
} else
{
var nShipSelectionView:ShipSelectionView =
ShipSelectionView(_viewFactory.createView(ShipSelectionView));
_viewFactory.notify(nShipSelectionView);
nShipSelectionView.setUp(onShipSelected);
}
}

private function onShipSelected( ship:ShipVO ):void
{
if (_selectedShipButton)
{
presenter.assignShipToFleet(_selectedFleet, ship, _selectedShipButton.index);
_shipActionBtn.text = _remove;
}
}

private function popPaywall( e:MouseEvent = null ):void
{
CommonFunctionUtil.popPaywall();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function get typeUnique():Boolean { return false; }

[Inject]
public function set presenter( value:IFleetPresenter ):void { _presenter = value; }
public function get presenter():IFleetPresenter { return IFleetPresenter(_presenter); }

override public function destroy():void
{
presenter.removeTransactionListener(transactionUpdate);
presenter.removeListenerOnFleetUpdated(onFleetsUpdated);
```

```
if (_timer)
{
if (_timer.running)
_timer.stop();

_timer.removeEventListener(TimerEvent.TIMER, onRepairTick);
}

_timer = null;

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_repairComponent)
_repairComponent.destroy();

_repairComponent = null;

if (_repairInProgressComponent)
_repairInProgressComponent.destroy();

_repairInProgressComponent = null;

if (_repairCostComponent)
_repairCostComponent.destroy();

_repairCostComponent = null;

if (_changeFleetNameBtn)
_changeFleetNameBtn.destroy();

_changeFleetNameBtn = null;

if (_docksBtnTwo)
_docksBtnTwo.destroy();

_docksBtnTwo = null;

if (_docksBtnOne)
_docksBtnOne.destroy();

_docksBtnOne = null;

if (_fleetSelection)
_fleetSelection.destroy();

_fleetSelection
```

```
= null;

var len:uint = _shipButtons.length;
for (var i:uint = 0; i < len; ++i)
{
_shipButtons[i].destroy();
_shipButtons[i] = null;
}
_shipButtons.length = 0;

if (_subTitle)
_subTitle.destroy();

_subTitle = null;

if (_fleetRatingText)
_fleetRatingText.destroy();

_fleetRatingText = null;

if (_fleetName)
_fleetName.destroy();

_fleetName = null;

if (_fleetDpsLabel)
_fleetDpsLabel.destroy();

_fleetDpsLabel = null;

if (_fleetHealthLabel)
_fleetHealthLabel.destroy();

_fleetHealthLabel = null;

if (_fleetCargoLabel)
_fleetCargoLabel.destroy();

_fleetCargoLabel = null;

if (_fleetMapSpeedLabel)
_fleetMapSpeedLabel.destroy();

_fleetMapSpeedLabel = null;

if (_fleetDpsValue)
_fleetDpsValue.destroy();

_fleetDpsValue = null;

if
```

```
(_fleetHealthValue)
_fleetHealthValue.destroy();

_fleetHealthValue = null;

if (_fleetCargoValue)
_fleetCargoValue.destroy();

_fleetCargoValue = null;

if (_fleetMapSpeedValue)
_fleetMapSpeedValue.destroy();

_fleetMapSpeedValue = null;

if (_powerUsedText)
_powerUsedText.destroy();

_powerUsedText = null;

if (_statusText)
_statusText.destroy();

_statusText = null;

if (_statusInfoText)
_statusInfoText.destroy();

_statusInfoText = null;

_repairTransaction = null;

if (_fleetDpsBar)
_fleetDpsBar.destroy();

_fleetDpsBar = null;

if (_fleetHealthBar)
_fleetHealthBar.destroy();

_fleetHealthBar = null;

if (_fleetCargoBar)
_fleetCargoBar.destroy();

_fleetCargoBar = null;

if (_fleetMapSpeedBar)
_fleetMapSpeedBar.destroy();

_fleetMapSpeedBar
```

```
= null;

if (_fleetNameLabel)
_fleetNameLabel.destroy();

_fleetNameLabel = null;

if (_fleetRatingLabel)
_fleetRatingLabel.destroy();

_fleetRatingLabel = null;

if (_fleetPowerLabel)
_fleetPowerLabel.destroy();

_fleetPowerLabel = null;

super.destroy();

}
}
}



-----------------------------------
File 805: igw\com\ui\modal\dock\FleetButton.as
package com.ui.modal.dock
{
import com.enum.FleetStateEnum;
import com.enum.ui.ButtonEnum;
import com.model.fleet.FleetVO;
import com.ui.UIFactory;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.DisplayObject;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.greensock.TweenLite;

public class FleetButton extends Sprite
{
private
```

```
var _fleet:FleetVO;

private var _index:int;

private var _bg:BitmapButton;

private var _queueBtn:BitmapButton;

private var _image:ImageComponent;

private var _statusText:Label;

private var _bgMask:Bitmap;

private var _healthBar:ProgressBar;

private var _battleSymbol:Bitmap;
private var _repairingSymbol:Bitmap;
private var _repairingSymbolBG:Bitmap;
private var _lockedSymbol:Bitmap;

private var _repairBGHolder:Sprite;

private var _isQueued:Boolean;
private var _isRepairing:Boolean;

public var onSelectFleet:Function;
public var onLoadShipImage:Function;

private var _docking:String = 'CodeString.Docks.FleetSelection.Docking'; //Docking
private var _out:String = 'CodeString.Docks.FleetSelection.Out'; //Out
private var _recalled:String = 'CodeString.Docks.FleetSelection.Recalled'; //Recalled
private var _notUnlocked:String = 'CodeString.Docks.FleetSelection.NotUnlocked'; //Not Yet
Unlocked!
private var _fleetLaunched:String = 'CodeString.Docks.FleetSelection.FleetIsLaunched'; //Fleet
Is Launched
private var _fleetDefendingText:String = 'CodeString.Alert.Battle.DefendingStatus'; //Defending

public function FleetButton()
{
super();

_bg = UIFactory.getButton(ButtonEnum.FLEET);
_bg.addEventListener(MouseEvent.CLICK, onMouse, false, 0, true);
_bg.selectable = true;

_image = new ImageComponent();
_image.mouseEnabled = false;
_image.init(_bg.width, _bg.height - 5);

_bgMask
```

```
= UIFactory.getBitmap('MaskBtnFleetBMD');
_bgMask.x = _bg.x;
_bgMask.y = _bg.y + 1;
_bgMask.cacheAsBitmap = true;

_healthBar = new ProgressBar();
_healthBar.init(ProgressBar.VERTICAL, new Bitmap(new BitmapData(_bg.width, _bg.height,
true, 0x8Cff0000)), null, 0.01);
_healthBar.x = _bg.x;
_healthBar.y = _bg.y;
_healthBar.mask = _bgMask;
_healthBar.cacheAsBitmap = true;
_healthBar.setMinMax(0, 1);
_healthBar.mouseChildren = false;
_healthBar.mouseEnabled = false;

_lockedSymbol = UIFactory.getBitmap('IconBlueLockedBMD');
_lockedSymbol.x = _bg.x + (_bg.width - _lockedSymbol.width) * 0.5;
_lockedSymbol.y = _bg.y + _bg.height - (_lockedSymbol.height + 5);
_lockedSymbol.visible = false;

_repairingSymbolBG = UIFactory.getBitmap('IconDockRepairCircleBMD');
_repairingSymbolBG.x = -_repairingSymbolBG.width * 0.5;
_repairingSymbolBG.y = -_repairingSymbolBG.height * 0.5;
_repairingSymbolBG.smoothing = true;

_repairBGHolder = new Sprite();
_repairBGHolder.addChild(_repairingSymbolBG);
_repairBGHolder.visible = false;
_repairBGHolder.x = _bg.x + (_repairBGHolder.width * 0.25);
_repairBGHolder.mouseChildren = false;
_repairBGHolder.mouseEnabled = false;

_repairingSymbol = UIFactory.getBitmap('IconDockRepairWrenchBMD');
_repairingSymbol.visible = false;
_repairingSymbol.x = _repairBGHolder.x - _repairingSymbol.width * 0.5;
_repairingSymbol.y = _repairBGHolder.y - _repairingSymbol.height * 0.5;

_queueBtn = UIFactory.getButton(ButtonEnum.QUEUE);
_queueBtn.x = _bg.x + _bg.width - _queueBtn.width;
_queueBtn.selectable = true;
_queueBtn.addEventListener(MouseEvent.CLICK, onQueueBtnClicked, false, 0, true);

_battleSymbol = UIFactory.getBitmap('CombatIconBMD');
_battleSymbol.visible = false;
_battleSymbol.x = _bg.x + (_bg.width - _battleSymbol.width) * 0.5;
_battleSymbol.y = _bg.y + (_bg.height - _battleSymbol.height) * 0.5;

_statusText = new Label(12, 0xffffff, 100, 20, true, 1);
_statusText.x = _healthBar.x;
_statusText.y
```

```
= _healthBar.y;
_statusText.width = _bg.width;
_statusText.height = _bg.height;
_statusText.multiline = true;
_statusText.constrictTextToSize = false;
_statusText.align = TextFormatAlign.CENTER;
_statusText.mouseEnabled = false;

addChild(_bg);
addChild(_image);
addChild(_healthBar);
addChild(_bgMask);
addChild(_battleSymbol);
addChild(_lockedSymbol);
addChild(_statusText);
addChild(_repairBGHolder);
addChild(_repairingSymbol);
addChild(_queueBtn);
}

private function onQueueBtnClicked( e:MouseEvent ):void
{
e.stopImmediatePropagation();
_isQueued = _queueBtn.selected;
}

public function showRepair( show:Boolean ):void
{
if (_bg.enabled)
{
_isRepairing = show;

if (_fleet.sector == '' && _fleet.currentHealth > 0)
_queueBtn.visible = !show;

if (_fleet != null)
{
_repairingSymbol.visible = show;
_repairBGHolder.visible = show;

if (show)
TweenLite.to(_repairBGHolder, 3000, {rotation:'540000', onUpdate:onRotate});
else
TweenLite.killTweensOf(_repairBGHolder);
}
}
}

private function onMouse( e:MouseEvent ):void
{
if
```

```
(_fleet != null && e.target != _queueBtn)
onSelectFleet(this);
}

override public function get height():Number
{
return _bg.height;
}

override public function get width():Number
{
return _bg.width;
}

public function get isQueued():Boolean
{
return _isQueued;
}

public function get fleet():FleetVO
{
return _fleet;
}

public function get id():String
{
return _fleet.id;
}

public function get index():int
{
return _index;
}

public function set selected( v:Boolean ):void
{
_bg.selected = v;
}

public function setFleet( fleet:FleetVO ):void
{
_fleet = fleet;
TweenLite.killTweensOf(_repairBGHolder);
if (_fleet == null || _fleet.numOfShips == 0)
{
if (_fleet == null)
{
_statusText.text = _notUnlocked;
_lockedSymbol.visible = true;
_bg.enabled = false;
}
```

```
else
_bg.enabled = true;
_image.clearBitmap();
_healthBar.amount = 0;
_queueBtn.visible = false;

} else
{
_bg.enabled = true;
_healthBar.amount = (1 - _fleet.currentHealth);
var asset:String = _fleet.asset;
if (asset != '')
onLoadShipImage(asset, onImageLoaded);

if (_isRepairing && _fleet.state != FleetStateEnum.REPAIRING)
showRepair(false);

if (_fleet.sector == '' && _fleet.currentHealth > 0 && !_isRepairing)
{
if (_fleet.state == FleetStateEnum.DEFENDING)
{
_statusText.text = _fleetDefendingText;
_queueBtn.visible = false;
} else
{
_statusText.text = '';
_queueBtn.visible = true;
}
} else if (_fleet.sector != '')
{
_queueBtn.visible = false;
if (_fleet.state != FleetStateEnum.DOCKING && _fleet.state !=
FleetStateEnum.FORCED_RECALLING)
_statusText.text = (_fleet.defendTarget != "") ? _fleetDefendingText : _out;
else
_statusText.text = _recalled;
} else
_queueBtn.visible = false;

if (_fleet.inBattle)
_battleSymbol.visible = true;
else
_battleSymbol.visible = false;

_lockedSymbol.visible = false;
}
}

private function onImageLoaded( asset:BitmapData ):void
{
if
```

```
(_image && _bg)
{
_image.onImageLoaded(asset);
_image.x = _bg.x + (_bg.width - _image.width) * 0.5;
_image.y = _bg.y + (_bg.height - _image.height) * 0.5;
}
}

private function onRotate():void
{
_repairingSymbolBG.smoothing = true;
}

public function set index( index:int ):void
{
_index = index;
}

public function destroy():void
{
if (_bg)
{
_bg.removeEventListener(MouseEvent.CLICK, onMouse);
_bg = UIFactory.destroyButton(_bg);
}
}
}
}

-------------------------------------
File 806: igw\com\ui\modal\dock\FleetSelection.as
package com.ui.modal.dock
{
import com.model.fleet.FleetVO;

import flash.display.Sprite;
import flash.utils.Dictionary;

import org.osflash.signals.Signal;

public class FleetSelection extends Sprite
{
private var _fleets:Dictionary;
private var _selectedFleet:FleetButton;
public var loadShipImage:Function;
public var onFleetSelected:Signal;

public function FleetSelection()
{
super();
_fleets
```

```
= new Dictionary();
onFleetSelected = new Signal(FleetButton);
}

public function addFleets( fleets:Vector.<FleetVO> ):void
{
var i:uint;
var len:uint = fleets.length;
var fleetIcon:FleetButton;
var currentFleetVO:FleetVO;
for (i = 0; i < 10; ++i)
{
currentFleetVO = (i < fleets.length) ? fleets[i] : null;
fleetIcon = new FleetButton();
fleetIcon.index = i;
fleetIcon.onSelectFleet = onSelected;
fleetIcon.onLoadShipImage = loadShipImage;
fleetIcon.setFleet(currentFleetVO);
addChild(fleetIcon);

if (currentFleetVO)
_fleets[currentFleetVO.id] = fleetIcon;
}

layout();
}

public function updateFleet( fleet:FleetVO, isRepairing:Boolean ):void
{
var i:uint;
var fleetBtn:FleetButton;

if (fleet.id in _fleets)
{
fleetBtn = _fleets[fleet.id];
fleetBtn.setFleet(fleet);
fleetBtn.showRepair(isRepairing);
}

layout();
}

public function setSelected( ID:String ):void
{
if (ID in _fleets)
{
var fleetIcon:FleetButton = _fleets[ID];
onSelected(fleetIcon);
} else if (ID == null)
{
onSelected(FleetButton(getChildAt(0)));
```

```
}
}

private function onSelected( selectedIcon:FleetButton ):void
{
if (_selectedFleet != null)
_selectedFleet.selected = false;

_selectedFleet = selectedIcon;

_selectedFleet.selected = true;

onFleetSelected.dispatch(selectedIcon);
}

private function onFleetClick( selectedFleet:FleetButton ):void
{
onSelected(selectedFleet);
}

public function layout():void
{
var fleetIcon:FleetButton;
var xPos:int = 0;
var yPos:int = 0;

var len:uint = numChildren;
for (var i:uint = 0; i < len; ++i)
{
fleetIcon = FleetButton(getChildAt(i));
fleetIcon.x = xPos;
fleetIcon.y = yPos;

xPos += fleetIcon.width + 12;
}
}

public function destroy():void
{
var len:uint = numChildren;
var fleetIcon:FleetButton;
for (var i:uint = 0; i < len; ++i)
{
fleetIcon = FleetButton(getChildAt(i));
fleetIcon.destroy();
fleetIcon = null;
}

_selectedFleet.destroy();
_selectedFleet
```

```
= null;

onFleetSelected.removeAll();
}

public function get queuedFleets():Array
{
var queuedFleets:Array = new Array();
var len:uint = numChildren;
var fleetBtn:FleetButton;
for (var i:uint = 0; i < len; ++i)
{
fleetBtn = FleetButton(getChildAt(i));
if (fleetBtn.isQueued)
queuedFleets.push(fleetBtn.fleet);
}
return queuedFleets;
}

public function getFleetBtn( id:String ):FleetButton
{
if (id in _fleets)
{
var fleetBtn:FleetButton = _fleets[id];
return fleetBtn;
}

return null;
}
}
}

-------------------------------------
File 807: igw\com\ui\modal\dock\ShipButton.as
package com.ui.modal.dock
{
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.utils.getDefinitionByName;

import org.greensock.TweenLite;
import org.osflash.signals.Signal;

public
```

```
class ShipButton extends ShipIcon
{
private var _isRepairing:Boolean;

private var _flagshipSymbol:Bitmap;
private var _repairingSymbol:Bitmap;
private var _repairingSymbolBG:Bitmap;

private var _repairBGHolder:Sprite;

public var onSelectShip:Signal;

private var TOP_LEFT_X_OFFSET:int = 32;

public function ShipButton()
{
super();
onSelectShip = new Signal(ShipButton, Boolean);

_repairingSymbolBG = UIFactory.getBitmap('IconDockRepairCircleBMD');
_repairingSymbolBG.x = -_repairingSymbolBG.width * 0.5;
_repairingSymbolBG.y = -_repairingSymbolBG.height * 0.5;
_repairingSymbolBG.smoothing = true;

_repairBGHolder = new Sprite();
_repairBGHolder.addChild(_repairingSymbolBG);
_repairBGHolder.visible = false;
_repairBGHolder.x = _bitmap.x + TOP_LEFT_X_OFFSET + (_repairBGHolder.width * 0.25)

_repairingSymbol = UIFactory.getBitmap('IconDockRepairWrenchBMD');
_repairingSymbol.visible = false;
_repairingSymbol.x = _repairBGHolder.x - _repairingSymbol.width * 0.5;
_repairingSymbol.y = _repairBGHolder.y - _repairingSymbol.height * 0.5;

_flagshipSymbol = UIFactory.getBitmap('DockShipBtnFlagshipBMD');
_flagshipSymbol.visible = false;
_flagshipSymbol.x = _bitmap.x + _bitmap.width * 0.75 - (TOP_LEFT_X_OFFSET + 22);
_flagshipSymbol.y = _bitmap.y - 13;

addChild(_flagshipSymbol);
addChild(_repairBGHolder);
addChild(_repairingSymbol);
}

override protected function onMouse( e:MouseEvent ):void
{
super.onMouse(e);
if (mouseEnabled)
{
switch (e.type)
{
```

```
case MouseEvent.CLICK:
if (!_fleet || _fleet.sector != '' || _isRepairing)
return;

onSelectShip.dispatch(this, true);
break;
}

}
}

public function getTooltip():String
{
var tooltip:String = '';
if (_ship)
tooltip = _ship.tooltip;

return tooltip;
}

override protected function _setShip( ship:ShipVO, fleet:FleetVO ):void
{
super._setShip(ship, fleet);
_repairingSymbol.visible = _repairBGHolder.visible = _isRepairing = false;
TweenLite.killTweensOf(_repairBGHolder);
}

public function showRepair( show:Boolean ):void
{

_isRepairing = show;

if (enabled)
{
if (_ship != null && _ship.currentHealth != 1)
{
_repairingSymbol.visible = _isRepairing;
_repairBGHolder.visible = _isRepairing;

if (show)
TweenLite.to(_repairBGHolder, 3000, {rotation:'540000', onUpdate:onRotate});
else
TweenLite.killTweensOf(_repairBGHolder);
}
}
}

private function onRotate():void
{
_repairingSymbolBG.smoothing
```

```
= true;
}

override public function set index( index:int ):void
{
super.index = index;
if (_index == 0)
_flagshipSymbol.visible = true;
}

override public function destroy():void
{
TweenLite.killTweensOf(_repairBGHolder);

onSelectShip.removeAll();
onSelectShip = null;

_flagshipSymbol = null;
super.destroy();
}

}
}


-----------------------------------
File 808: igw\com\ui\modal\dock\ShipIcon.as
package com.ui.modal.dock
{
import com.model.fleet.FleetVO;
import com.model.fleet.ShipVO;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.ui.UIFactory;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.misc.ImageComponent;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.utils.getDefinitionByName;

import org.osflash.signals.Signal;

public class ShipIcon extends BitmapButton
{
protected var _index:int;
protected var _fleet:FleetVO;
protected var _ship:ShipVO;

private
```

```
var _image:ImageComponent;
private var _bgMask:Bitmap;

private var _healthBar:ProgressBar;
private var _isSelected:Boolean;

public var onLoadShipImage:Signal;

public function ShipIcon()
{
super.init(UIFactory.getBitmapData('BtnShipUpBMD'),
UIFactory.getBitmapData('BtnShipROBMD'), UIFactory.getBitmapData('BtnShipDownBMD'),
UIFactory.getBitmapData('BtnShipDisabledBMD'), UIFactory.
getBitmapData('BtnShipSelectedBMD'));

_image = new ImageComponent();
_image.init(_bitmap.width, _bitmap.height);

_bgMask = UIFactory.getBitmap('IconShipDamagedBMD');
_bgMask.x = _bitmap.x + 3;
_bgMask.y = _bitmap.y + 3;
_bgMask.cacheAsBitmap = true;

_healthBar = new ProgressBar();
_healthBar.init(ProgressBar.VERTICAL, new Bitmap(new BitmapData(_bitmap.width,
_bitmap.height, true, 0x8Cff0000)), null, 0.01);
_healthBar.x = _bitmap.x;
_healthBar.y = _bitmap.y;
_healthBar.mask = _bgMask;
_healthBar.cacheAsBitmap = true;
_healthBar.setMinMax(0, 1);

addChild(_image);
addChild(_healthBar);
addChild(_bgMask);

onLoadShipImage = new Signal(String, Function);

mouseChildren = false;
}

public function setBarValue( value:Number ):void
{
_healthBar.amount = value;
_healthBar.visible = _healthBar.amount != 0;
}

public function setShip( ship:ShipVO, prototype:IPrototype = null, fleet:FleetVO = null ):void
{
if (ship)
_setShip(ship,
```

```
fleet);
else if (prototype)
{
var vo:ShipVO = new ShipVO();
vo.prototypeVO = prototype;
_setShip(vo, fleet);
} else
_setShip(ship, fleet);
}

protected function _setShip( ship:ShipVO, fleet:FleetVO ):void
{
_fleet = fleet;
_ship = ship;
if (!_ship)
{
_image.clearBitmap();
_healthBar.amount = 0;
_healthBar.visible = false;
} else
{
onLoadShipImage.dispatch(_ship.prototypeVO.asset, onImageLoaded);
var isNewRepairSystemActive:Boolean =
PrototypeModel.instance.getConstantPrototypeValueByName("isNewRepairSystemActive");
if (isNewRepairSystemActive)
{
if(fleet)
{
var newHealthPct:Number = fleet.GetFleetHealthFromRepairTimeRemaining();
var pctDamageRemaining:Number;
if(fleet.currentHealth != 1)
pctDamageRemaining = (1 - newHealthPct) / (1 - fleet.currentHealth);
else
pctDamageRemaining = 0;

_healthBar.amount = ( ( 1- _ship.currentHealth) * pctDamageRemaining );
}
}
else
_healthBar.amount = (1 - _ship.currentHealth);

_healthBar.visible = _healthBar.amount != 0;
}
}

public function onImageLoaded( asset:BitmapData ):void
{
if (_image)
{
_image.onImageLoaded(asset);
_image.smoothing
```

```
= true;
_image.x = _bitmap.x + (_bitmap.width - _image.width) * 0.5;
_image.y = _bitmap.y + (_bitmap.height - _image.height) * 0.5;
}
}

public function clearImageBitmap():void
{
_image.clearBitmap();
}

public function set index( index:int ):void
{
_index = index;
}

public function get index():int
{
return _index;
}

public function get ship():ShipVO
{
return _ship;
}

public function get id():String
{
return _ship.id;
}

override public function set scaleX( value:Number ):void
{
_bitmap.scaleX = value;
_bgMask.scaleX = value;
_healthBar.scaleX = value;
_bgMask.cacheAsBitmap = true;
_healthBar.mask = _bgMask;
_healthBar.cacheAsBitmap = true;
}

override public function get scaleX():Number
{
return _bitmap.scaleX;
}

override public function set scaleY( value:Number ):void
{
_bitmap.scaleY = value;
_bgMask.scaleY = value;
_healthBar.scaleY
```

```
= value;

_bgMask.cacheAsBitmap = true;
_healthBar.mask = _bgMask;
_healthBar.cacheAsBitmap = true;

}

public function scale( x:Number, y:Number ):void
{
_bitmap.scaleX = x;
_bitmap.scaleY = y;

_bgMask.scaleX = y;
_bgMask.scaleY = x;

_healthBar.scaleX = x;
_healthBar.scaleY = y;

_healthBar.x = _bitmap.x;
_healthBar.y = _bitmap.y;
_bgMask.x = _bitmap.x + 1;
_bgMask.y = _bitmap.y + 1;

_bgMask.cacheAsBitmap = true;
_healthBar.mask = _bgMask;
_healthBar.cacheAsBitmap = true;
}

override public function get scaleY():Number
{
return _bitmap.scaleY;
}

override public function destroy():void
{
super.destroy();

if (_healthBar)
{
_healthBar.destroy();
_healthBar = null;
}
}
}
}

-----------------------------------
File 809: igw\com\ui\modal\dock\ShipPanelSelection.as
package com.ui.modal.dock
{
```

```
import com.model.fleet.ShipVO;
import com.ui.UIFactory;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;
import com.ui.core.ScaleBitmap;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.filters.GlowFilter;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class ShipPanelSelection extends Sprite
{
public var onClicked:Signal;

private var _ship:ShipVO;
private var _shipBG:Bitmap;
private var _shipName:Label;
private var _shipCustomName:Label;

private var _dpsBG:Bitmap;
private var _armorBG:Bitmap;
private var _rangeBG:Bitmap;
private var _evasionBG:Bitmap;
private var _maskingBG:Bitmap;
private var _cargoBG:Bitmap;

private var _health:Label;
private var _healthTitle:Label;
private var _dps:Label;
private var _armor:Label;
private var _range:Label;
private var _evasion:Label;
private var _masking:Label;
private var _cargo:Label;

private var _healthBar:ProgressBar;

private
```

```
var _shipBtn:BitmapButton;

private var _image:ImageComponent;

public function ShipPanelSelection()
{
addEventListener(MouseEvent.CLICK, onMouse, false, 5, true);

var statsBGBitmapData:BitmapData = new BitmapData(153, 18, false, 0x437b89);

_shipBtn = new BitmapButton();
_shipBtn.init(UIFactory.getBitmapData('SelectFrameBMD'),
UIFactory.getBitmapData('SelectFrameRollOverBMD'),
UIFactory.getBitmapData('SelectFrameDownBMD'));
_shipBtn.scale9Grid = new Rectangle(10, 10, 10, 10);
_shipBtn.setSize(600, 113);

onClicked = new Signal(ShipVO);

_shipBG = UIFactory.getBitmap('SelectionFrameBMD');
_shipBG.x = 3;
_shipBG.y = 7;

_image = new ImageComponent();
_image.init(_shipBG.width, _shipBG.height);

_shipCustomName = new Label(18, 0xffffff, 90, 20, true);
_shipCustomName.autoSize = TextFieldAutoSize.LEFT;
_shipCustomName.allCaps = true;
_shipCustomName.x = 115;
_shipCustomName.y = 20;
_shipCustomName.constrictTextToSize = false;

_shipName = new Label(12, 0xffffff, 90, 20, true);
_shipName.autoSize = TextFieldAutoSize.LEFT;
_shipName.allCaps = true;
_shipName.x = 115;
_shipName.y = 5;
_shipName.constrictTextToSize = false;

_dpsBG = new Bitmap(statsBGBitmapData);
_dpsBG.alpha = .5;
_dpsBG.x = 117;
_dpsBG.y = 45;

_armorBG = new Bitmap(statsBGBitmapData);
_armorBG.alpha = .5;
_armorBG.x = 117;
_armorBG.y = 70;

_rangeBG
```

```
= new Bitmap(statsBGBitmapData);
_rangeBG.alpha = .5;
_rangeBG.x = 438;
_rangeBG.y = 45;

_evasionBG = new Bitmap(statsBGBitmapData);
_evasionBG.alpha = .5;
_evasionBG.x = 277;
_evasionBG.y = 45;

_maskingBG = new Bitmap(statsBGBitmapData);
_maskingBG.alpha = .5;
_maskingBG.x = 277;
_maskingBG.y = 70;

_cargoBG = new Bitmap(statsBGBitmapData);
_cargoBG.alpha = .5;
_cargoBG.x = 438;
_cargoBG.y = 70;

_dps = new Label(12, 0xffffff, _dpsBG.width, _dpsBG.height, true, 1);
_dps.align = TextFormatAlign.LEFT;
_dps.x = _dpsBG.x;
_dps.y = _dpsBG.y;
_dps.constrictTextToSize = false;

_armor = new Label(12, 0xffffff, _armorBG.width, _armorBG.height, true, 1);
_armor.align = TextFormatAlign.LEFT;
_armor.x = _armorBG.x;
_armor.y = _armorBG.y;
_armor.constrictTextToSize = false;

_range = new Label(12, 0xffffff, _rangeBG.width, _rangeBG.height, true, 1);
_range.align = TextFormatAlign.LEFT;
_range.x = _rangeBG.x;
_range.y = _rangeBG.y;
_range.constrictTextToSize = false;

_evasion = new Label(12, 0xffffff, _evasionBG.width, _evasionBG.height, true, 1);
_evasion.align = TextFormatAlign.LEFT;
_evasion.x = _evasionBG.x;
_evasion.y = _evasionBG.y;
_evasion.constrictTextToSize = false;

_masking = new Label(12, 0xffffff, _maskingBG.width, _maskingBG.height, true, 1);
_masking.align = TextFormatAlign.LEFT;
_masking.x = _maskingBG.x;
_masking.y = _maskingBG.y;
_masking.constrictTextToSize = false;

_cargo
```

```
= new Label(12, 0xffffff, _cargoBG.width, _cargoBG.height, true, 1);
_cargo.align = TextFormatAlign.LEFT;
_cargo.x = _cargoBG.x;
_cargo.y = _cargoBG.y;
_cargo.constrictTextToSize = false;

var healthy:ScaleBitmap = PanelFactory.getScaleBitmapPanel('FleetHealthBarGoodBMD', 140,
21, new Rectangle(2, 10, 1, 1));
var damaged:ScaleBitmap = PanelFactory.getScaleBitmapPanel('FleetHealthBarHurtBMD', 140,
20, new Rectangle(2, 10, 1, 1));
_healthBar = new ProgressBar()
_healthBar.init(ProgressBar.HORIZONTAL, healthy, damaged, 0.01);
_healthBar.x = 451;
_healthBar.y = 10;
_healthBar.setMinMax(0, 1);

_health = new Label(12, 0xffffff, _healthBar.width, 25, true, 1);
_health.align = TextFormatAlign.CENTER;
_health.height -= 10;
_health.x = _healthBar.x;
_health.y = _healthBar.y;
_health.constrictTextToSize = false;

_healthTitle = new Label(16, 0xffffff, 120, 25, true, 1);
_healthTitle.align = TextFormatAlign.LEFT;
_healthTitle.constrictTextToSize = false;
_healthTitle.text = 'CodeString.Shared.HealthTitle';
_healthTitle.x = _healthBar.x - _healthTitle.textWidth - 5;
_healthTitle.y = _healthBar.y - 3;

addChild(_shipBtn);
addChild(_shipName);
addChild(_shipCustomName);
addChild(_shipBG);
addChild(_dpsBG);
addChild(_armorBG);
addChild(_rangeBG);
addChild(_evasionBG);
addChild(_maskingBG);
addChild(_cargoBG);
addChild(_dps);
addChild(_armor);
addChild(_range);
addChild(_evasion);
addChild(_masking);
addChild(_cargo);
addChild(_healthBar);
addChild(_healthTitle);
addChild(_health);
addChild(_image);
}
```

```
public function onLoadImage( asset:BitmapData ):void
{
if (_image)
{
_image.onImageLoaded(asset);
_image.x = _shipBG.x + (_shipBG.width - _image.width) * 0.5
_image.y = _shipBG.y + (_shipBG.height - _image.height) * 0.5
}
}

public function setName( name:String ):void
{
_shipName.text = name;
}
public function setCustomName( name:String ):void
{
if(name.length>0)
_shipCustomName.text = name;
else
{
_shipCustomName.text = _shipName.text;
_shipName.text = "";
}

}

public function set ship( ship:ShipVO ):void
{
_ship = ship;

_dps.setTextWithTokens('CodeString.Shared.Damage', {'[[Number.Damage]]':_ship.shipDps});

_armor.setTextWithTokens('CodeString.Shared.Armor', {'[[Number.Armor]]':_ship.armor});


_range.setTextWithTokens('CodeString.Shared.Range', {'[[Number.Range]]':_ship.maxRange});

_evasion.setTextWithTokens('CodeString.Shared.Evasion',
{'[[Number.Evasion]]':_ship.evasion});

_masking.setTextWithTokens('CodeString.Shared.Masking',
{'[[Number.Masking]]':_ship.masking});

_cargo.setTextWithTokens('CodeString.Shared.Cargo', {'[[Number.Cargo]]':_ship.cargo});

_healthBar.amount = _ship.currentHealth;

var totalHealth:int = _ship.healthAmount;

_health.text
```

```
= Math.round(_ship.currentHealth * totalHealth) + ' / ' + totalHealth;


var rarity:String = _ship.getValue('rarity');
_shipName.textColor = CommonFunctionUtil.getRarityColor(rarity);
_shipCustomName.textColor = _shipName.textColor;
if (rarity != 'Common')
_shipBG.filters = [CommonFunctionUtil.getRarityGlow(rarity)];
}

private function createGlow( rgb:uint ):GlowFilter
{
var glow:GlowFilter = new GlowFilter()
glow.inner = true;
glow.color = rgb;
glow.blurX = 5;
glow.blurY = 5;

return glow;
}

public function get itemClass():String
{
return _ship.getValue('itemClass');
}

public function getTooltip():String
{
var tooltip:String = '';
if (_ship)
tooltip = _ship.tooltip;

return tooltip;
}

private function onMouse( e:MouseEvent ):void
{
onClicked.dispatch(_ship);
}

public function destroy():void
{
removeEventListener(MouseEvent.CLICK, onMouse);

if (onClicked)
onClicked.removeAll();

onClicked = null;

_shipBG = null;
_dpsBG
```

```
= null;
_armorBG = null;
_rangeBG = null;
_evasionBG = null;
_maskingBG = null;
_cargoBG = null;

if (_shipName)
_shipName.destroy();

_shipName = null;

if (_shipCustomName)
_shipCustomName.destroy();

_shipCustomName = null;

if (_health)
_health.destroy();

_health = null;

if (_healthTitle)
_healthTitle.destroy();

_healthTitle = null;

if (_dps)
_dps.destroy();

_dps = null;

if (_armor)
_armor.destroy();

_armor = null;

if (_range)
_range.destroy();

_range = null;

if (_evasion)
_evasion.destroy();

_evasion = null;

if (_masking)
_masking.destroy();

_masking
```

```
= null;

if (_cargo)
_cargo.destroy();

_cargo = null;


if (_healthBar)
ObjectPool.give(_healthBar);

_healthBar = null;

if (_shipBtn)
ObjectPool.give(_shipBtn);

_shipBtn = null;

if (_image)
ObjectPool.give(_image);

_image = null;
}

}
}

-------------------------------------
File 810: igw\com\ui\modal\dock\ShipSelectionView.as
package com.ui.modal.dock
{
import com.enum.FactionEnum;
import com.model.fleet.ShipVO;
import com.model.player.CurrentUser;
import com.presenter.starbase.IFleetPresenter;
import com.ui.core.component.label.Label;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.filterlist.FilterList;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import
```

```
flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

public class ShipSelectionView extends View
{
private var _bg:ScaleBitmap;
private var _closeBtn:BitmapButton;
private var _topBG:Bitmap;
private var _shipPanelSelections:Vector.<ShipPanelSelection>;
private var _visiblePanelSelections:Vector.<ShipPanelSelection>;
private var _onShipSelected:Function;

private var _title:Label;
private var _scrollbar:VScrollbar;
private var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _holder:Sprite;
private var _hitBlocker:Sprite;

private var _filterList:FilterList;

private var PADDING:int = 45;

private var _titleText:String = 'CodeString.Shipyard.SelectionTitle';

[Inject]
public var tooltip:Tooltips;

public function ShipSelectionView()
{
super();
}

[PostConstruct]
override public function init():void
{
super.init();

_topBG = UIFactory.getBitmap('ShipSelectHeaderBMD');

var bgRect:Rectangle = new Rectangle(0, 40, 637, 40);
_bg = UIFactory.getScaleBitmap('ShipSelectFrameBMD');
_bg.scale9Grid = bgRect;
_bg.y = _topBG.height + 3;
_bg.width = 640;
_bg.height = 515;

_shipPanelSelections = new Vector.<ShipPanelSelection>;
_visiblePanelSelections
```

```
= new Vector.<ShipPanelSelection>;

_holder = new Sprite();
_holder.x = _bg.x + 8;
_holder.y = _bg.y + 8;
_maxHeight = 0;

_scrollRect = new Rectangle(_bg.x, _bg.y, _bg.width - 10, _bg.height - 30);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect

_hitBlocker = new Sprite();
_hitBlocker.graphics.clear();
_hitBlocker.graphics.lineStyle(2, 0xffffff, 0);
_hitBlocker.graphics.beginFill(0xffffff, 0);
_hitBlocker.graphics.moveTo(0, 0);
_hitBlocker.graphics.lineTo(_bg.width, 0);
_hitBlocker.graphics.lineTo(_bg.width, _bg.y + _bg.height);
_hitBlocker.graphics.lineTo(0, _bg.y + _bg.height);
_hitBlocker.graphics.lineTo(0, 0);
_hitBlocker.graphics.endFill();


_title = new Label(15, 0xffffffff, 90, 20, true);
_title.autoSize = TextFieldAutoSize.LEFT;
_title.allCaps = true;
_title.x = 12;
_title.y = 6;
_title.constrictTextToSize = false;
_title.text = _titleText;

var faction:String;
switch (CurrentUser.faction)
{
case FactionEnum.IGA:
faction = 'IGA';
break;
case FactionEnum.SOVEREIGNTY:
faction = 'SOV';
break;
case FactionEnum.TYRANNAR:
faction = 'TY';
break;
}

_filterList = new FilterList();
_filterList.init('', false, true, true);
_filterList.x = 37;
_filterList.y = 39;
_filterList.onSelectionChanged.add(filterShips);

var
```

```
fighterBtn:BitmapButton;
var heavyFighterBtn:BitmapButton;
var corvetteBtn:BitmapButton;
var destroyerBtn:BitmapButton;
var battleshipBtn:BitmapButton;
var dreadnoughtBtn:BitmapButton;
var transportBtn:BitmapButton;

fighterBtn = ButtonFactory.getBitmapButton(faction + 'LFBMD', 0, 0, '', 0xFFFFFF, faction +
'LFRollOverBMD', null, null, faction + 'LFRollOverBMD');
_filterList.addFilterBtn('Fighter', fighterBtn, 45);

heavyFighterBtn = ButtonFactory.getBitmapButton(faction + 'HFBMD', 0, 0, '', 0xFFFFFF, faction
+ 'HFRollOverBMD', null, null, faction + 'HFRollOverBMD');
_filterList.addFilterBtn('Heavy Fighter', heavyFighterBtn, 45);

corvetteBtn = ButtonFactory.getBitmapButton(faction + 'CVBMD', 0, 0, '', 0xFFFFFF, faction +
'CVRollOverBMD', null, null, faction + 'CVRollOverBMD');
_filterList.addFilterBtn('Corvette', corvetteBtn, 45);

destroyerBtn = ButtonFactory.getBitmapButton(faction + 'DDBMD', 0, 0, '', 0xFFFFFF, faction +
'DDRollOverBMD', null, null, faction + 'DDRollOverBMD');
_filterList.addFilterBtn('Destroyer', destroyerBtn, 45);

battleshipBtn = ButtonFactory.getBitmapButton(faction + 'BSBMD', 0, 0, '', 0xFFFFFF, faction +
'BSRollOverBMD', null, null, faction + 'BSRollOverBMD');
_filterList.addFilterBtn('Battleship', battleshipBtn, 45);

dreadnoughtBtn = ButtonFactory.getBitmapButton(faction + 'DNBMD', 0, 0, '', 0xFFFFFF, faction
+ 'DNRollOverBMD', null, null, faction + 'DNRollOverBMD');
_filterList.addFilterBtn('Dreadnought', dreadnoughtBtn, 45);

transportBtn = ButtonFactory.getBitmapButton(faction + 'TSBMD', 0, 0, '', 0xFFFFFF, faction +
'TSRollOverBMD', null, null, faction + 'TSRollOverBMD');
_filterList.addFilterBtn('Transport', transportBtn, 45);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 35, 35);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _bg.x + _bg.width - 19;
var scrollbarYPos:Number = _bg.y + 15;
_scrollbar.init(7, _scrollRect.height - 15, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 30;

addChild(_hitBlocker);
```

```
addChild(_topBG)
addChild(_bg);
addChild(_closeBtn);
addChild(_filterList);
addChild(_holder);
addChild(_scrollbar);
addChild(_title);

addEffects();
effectsIN();
}

override public function get height():Number
{
return _hitBlocker.height;
}

override public function get width():Number
{
return _hitBlocker.width;
}

private function filterShips( filters:Array ):void
{
if (_holder.numChildren > 0)
_holder.removeChildren(0, (_holder.numChildren - 1));

_visiblePanelSelections.length = 0;
presenter.shipSelectionFilter = filters;
var len:uint = _shipPanelSelections.length;
var currentShipPanelSelection:ShipPanelSelection;
for (var i:uint = 0; i < len; ++i)
{
currentShipPanelSelection = _shipPanelSelections[i];
if (filters == null || filters.length == 0 || filters.indexOf(currentShipPanelSelection.itemClass) != -1)
{
_holder.addChild(currentShipPanelSelection);
_visiblePanelSelections.push(currentShipPanelSelection);
}
}

layout();
_scrollbar.resetScroll();
}

override public function set visible( value:Boolean ):void
{
super.visible = value;
_scrollbar.resetScroll();
}
```

```
private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

public function addShips( ships:Vector.<ShipVO> ):void
{

var selectionsLen:uint = _shipPanelSelections.length;
var len:uint = ships.length;
var currentShipVO:ShipVO;
var currentShipPanelSelection:ShipPanelSelection;
var i:uint = 0
var currentShipCount:uint = 0;
for (; i < len; ++i)
{
currentShipVO = ships[i];

if (currentShipVO.built != true)
continue;

if (currentShipCount < selectionsLen)
{
currentShipPanelSelection = _shipPanelSelections[currentShipCount];
++currentShipCount;
} else
{
currentShipPanelSelection = new ShipPanelSelection();
currentShipPanelSelection.onClicked.add(onShipClick);
_shipPanelSelections.push(currentShipPanelSelection);
}

currentShipPanelSelection.ship = currentShipVO;
tooltip.addTooltip(currentShipPanelSelection, this, currentShipPanelSelection.getTooltip, '', 250,
180, 14);

if (currentShipVO.prototypeVO != null)
{
presenter.loadIconFromEntityData(currentShipVO.prototypeVO.asset,
currentShipPanelSelection.onLoadImage);
presenter.getProtoTypeUIName(currentShipVO.prototypeVO,
currentShipPanelSelection.setName);
}
currentShipPanelSelection.setCustomName(currentShipVO.shipName);

}

if
```

```
(i < selectionsLen)
{
var startValue:uint = i;
for (; i < selectionsLen; ++i)
{
currentShipPanelSelection = _shipPanelSelections[i];
currentShipPanelSelection.destroy();
currentShipPanelSelection = null;
}
_shipPanelSelections.splice(startValue, selectionsLen - startValue);
}

if (presenter.shipSelectionFilter != null && presenter.shipSelectionFilter.length > 0)
_filterList.selectFilterByFilter(presenter.shipSelectionFilter);
else
filterShips(null);
}

private function onShipClick( ship:ShipVO ):void
{
if (ship)
{
_onShipSelected(ship);
destroy();
}
}

public function setUp( onShipSelected:Function ):void
{
_onShipSelected = onShipSelected
addShips(presenter.unassignedShips);
}

private function layout():void
{
var len:uint = _visiblePanelSelections.length;
var shipSelection:ShipPanelSelection;
var yPos:int = 15;
_maxHeight = 0;
var shipCount:int;
for (var i:uint = 0; i < len; ++i)
{
shipSelection = _visiblePanelSelections[i];
shipSelection.x = 10;
shipSelection.y = yPos;
_maxHeight += shipSelection.height + 5;
yPos += shipSelection.height + 5;
++shipCount;
}
_maxHeight += 10;

_scrollbar.updateScrollableHeight(_maxHeight);
```

```
}

override public function get typeUnique():Boolean { return false; }

[Inject]
public function set presenter( value:IFleetPresenter ):void { _presenter = value; }
public function get presenter():IFleetPresenter { return IFleetPresenter(_presenter); }

override public function destroy():void
{
super.destroy();

if (_holder.numChildren > 0)
_holder.removeChildren(0, (_holder.numChildren - 1));

_visiblePanelSelections.length = 0;

var len:uint = _shipPanelSelections.length;
var currentShipPanel:ShipPanelSelection;
for (var i:uint = 0; i < len; ++i)
{
currentShipPanel = _shipPanelSelections[i];
currentShipPanel.destroy();
currentShipPanel = null;
}
_shipPanelSelections.length = 0;
_closeBtn.destroy();
_closeBtn = null;

if (_title)
_title.destroy();

_title = null;
}
}

}


-----------------------------------
File 811: igw\com\ui\modal\event\EventReward.as
package com.ui.modal.event
{
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.IEventPresenter;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import
```

```
com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BlendMode;
import flash.display.Sprite;
import flash.events.TimerEvent;
import flash.filters.GlowFilter;
import flash.geom.Matrix;
import flash.geom.Point;
import flash.text.TextFormatAlign;
import flash.utils.Timer;

import org.adobe.utils.StringUtil;
import org.greensock.TweenLite;
import org.greensock.easing.Linear;
import org.shared.ObjectPool;

public class EventReward extends Sprite
{
private var _reward:IPrototype;
private var _index:uint;
private var _scoreRequirement:uint;
private var _unlocked:Boolean;
private var _isNextRewardDisplay:Boolean;

private var _image:ImageComponent;
private var _imageFrame:ScaleBitmap;
private var _maskFrame:ScaleBitmap;

private var _checkmark:Bitmap;
private var _lock:Bitmap;

private var _delayTimer:Timer;

private var _gradient:Sprite;

private var _scoreRequirementText:Label;

private var _tooltips:Tooltips;
private var _presenter:IEventPresenter;

public function EventReward( width:Number, height:Number, isNextRewardDisplay:Boolean,
unlocked:Boolean, tooltips:Tooltips, presenter:IEventPresenter )
{
super();
_presenter = presenter;
_tooltips = tooltips;
_unlocked
```

```
= unlocked;
_isNextRewardDisplay = isNextRewardDisplay;

_image = ObjectPool.get(ImageComponent);
_image.init(width, height);
_image.center = true;
_image.x = _image.y = 9;
_image.mouseEnabled = _image.mouseChildren = false;

_imageFrame = UIFactory.getPanel(PanelEnum.CHARACTER_FRAME, width, height, 9, 9);

addChild(_imageFrame);
addChild(_image);

if (!isNextRewardDisplay)
{
if (!unlocked)
_image.filters = [CommonFunctionUtil.getGreyScaleFilter()]

_lock = UIFactory.getBitmap("IconBlueLockedBMD");
_lock.x = _imageFrame.x + (width - _lock.width) * .5;
_lock.y = _imageFrame.y + (height - _lock.height) * .5;
_lock.visible = !unlocked;

_checkmark = UIFactory.getBitmap("EventCheckmarkBMD");
_checkmark.scaleX = _checkmark.scaleY = 0.65;
_checkmark.smoothing = true;
_checkmark.x = _imageFrame.x + _imageFrame.width - _checkmark.width * 0.85;
_checkmark.y = _imageFrame.y + _imageFrame.height - _checkmark.height;
_checkmark.visible = unlocked;

addChild(_lock);
addChild(_checkmark);
}

}

public function set reward( v:IPrototype ):void
{
_reward = v;

if (_reward)
{
visible = true;
var assetVO:AssetVO = _presenter.getAssetVO(_reward);
_presenter.loadIcon(assetVO.mediumImage, _image.onImageLoaded);


var rarity:String = _reward.getUnsafeValue('rarity');
if (rarity != 'Common')
{
```

```
var glow:GlowFilter = CommonFunctionUtil.getRarityGlow(rarity);
_imageFrame.filters = [glow];
} else
_imageFrame.filters = [];

_tooltips.addTooltip(this, null, null, StringUtil.getTooltip(_reward.getValue("type"), _reward, false,
null));
} else
visible = false;
}

public function set scoreRequirement( v:uint ):void
{
_scoreRequirement = v;

_scoreRequirementText = new Label(12, 0xf0f0f0, _imageFrame.width, 100, true, 1);
_scoreRequirementText.align = TextFormatAlign.CENTER;
_scoreRequirementText.constrictTextToSize = false;
_scoreRequirementText.multiline = true;
_scoreRequirementText.x = _imageFrame.x;
_scoreRequirementText.y = _imageFrame.y + _imageFrame.height;
_scoreRequirementText.text = StringUtil.commaFormatNumber(v);
addChild(_scoreRequirementText);
}

public function get scoreRequirement():uint { return _scoreRequirement; }

public function set active( v:Boolean ):void
{
if (!_isNextRewardDisplay)
{
addGradientEffect();
} else
{
_delayTimer = new Timer(1000, 1);
_delayTimer.addEventListener(TimerEvent.TIMER_COMPLETE, onDelayFinished, false, 0,
true);
_delayTimer.start();
}
}

private function addGradientEffect():void
{
_maskFrame = UIFactory.getPanel(PanelEnum.CHARACTER_FRAME, _imageFrame.width,
_imageFrame.height, 9, 9);
_maskFrame.cacheAsBitmap = true;

_gradient = new Sprite();
_gradient.graphics.beginBitmapFill(UIFactory.getBitmapData("ShineBMD"));
_gradient.graphics.drawRect(0,
```

```
0, _imageFrame.width * 4, _imageFrame.height * 4);
_gradient.graphics.endFill();
_gradient.blendMode = BlendMode.ADD;
_gradient.x = _imageFrame.x + (_imageFrame.width - _gradient.width) * 0.5;
_gradient.cacheAsBitmap = true;
_gradient.alpha = 0;
_gradient.mask = _maskFrame;

addChild(_gradient);
addChild(_maskFrame);

onMoveComplete();
}

private function onMoveComplete():void
{
_gradient.y = _imageFrame.y + (_imageFrame.height - _gradient.height) * 0.75;
TweenLite.to(_gradient, 0.5, {alpha:1, onComplete:onFadeInComplete, delay:2, overwrite:0});
TweenLite.to(_gradient, 1.5, {y:(_gradient.y + _gradient.height * 0.25),
onComplete:onMoveComplete, delay:2, overwrite:0, ease:Linear.easeNone});
}

private function onFadeInComplete():void
{
TweenLite.to(_gradient, 0.5, {alpha:0, delay:0.25, overwrite:0});
}

private function onDelayFinished( e:TimerEvent ):void
{
_delayTimer.removeEventListener(TimerEvent.TIMER_COMPLETE, onDelayFinished);
addGradientEffect();
}

public function set index( v:uint ):void
{
_index = v;
}

public function get index():uint
{
return _index;
}


override public function get height():Number { return _imageFrame.height; }
override public function get width():Number { return _imageFrame.width; }

public function destroy():void
{
if (_gradient)
TweenLite.killTweensOf(_gradient);
```

```
if (_delayTimer && _delayTimer.running)
{
_delayTimer.stop();
_delayTimer.removeEventListener(TimerEvent.TIMER_COMPLETE, onDelayFinished);
}

_delayTimer = null;
_gradient = null;
_lock = null;
_imageFrame = null;
_maskFrame = null;

if (_tooltips)
_tooltips.removeTooltip(this, null);

if (_image)
ObjectPool.give(_image);

_image = null;

if (_scoreRequirementText)
_scoreRequirementText.destroy();

_scoreRequirementText = null;
}
}
}

-------------------------------------
File 812: igw\com\ui\modal\event\EventView.as
package com.ui.modal.event
{
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.event.EventVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.presenter.shared.IEventPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.ProgressBar;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.util.CommonFunctionUtil;

import
```

```
flash.display.Bitmap;
import flash.display.BlendMode;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;

import org.greensock.TweenLite;
import org.shared.ObjectPool;

public class EventView extends View
{
private var _bg:DefaultWindowBG;

private var _eventTitle:Label;
private var _eventDescription:Label;
private var _eventObjective:Label;
private var _eventReward:Label;
private var _eventBuffs:Label;
private var _score:Label;

private var _eventImage:ImageComponent;

private var _eventProgressBar:ProgressBar;

private var _eventObjectiveContainer:ScaleBitmap;
private var _eventImageContainer:ScaleBitmap;
private var _eventRewardsContainer:ScaleBitmap;

private var _nextArrow:Bitmap;
private var _activeArrow:Sprite;

private var _nextReward:EventReward;

private var _eventDescriptionContainer:Sprite;
private var _eventRewards:Sprite;

private var _currentActiveEvent:EventVO;

private var _tooltips:Tooltips;

private var _titleText:String = 'EVENT';

[PostConstruct]
override public function init():void
{
super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(1020,
```

```
604);
_bg.addTitle(_titleText, 200);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_eventTitle = new Label(40, 0xf0f0f0, 482, 50);
_eventTitle.align = TextFormatAlign.CENTER;
_eventTitle.x = 46;
_eventTitle.y = 62;

_eventDescription = new Label(12, 0xf0f0f0, 464, 100, true, 1);
_eventDescription.autoSize = TextFieldAutoSize.LEFT
_eventDescription.align = TextFormatAlign.LEFT;
_eventDescription.constrictTextToSize = false;
_eventDescription.multiline = true;
_eventDescription.x = 55;
_eventDescription.y = 108;

_eventObjective = new Label(12, 0xf0f0f0, 956, 100, true, 1);
_eventObjective.align = TextFormatAlign.LEFT;
_eventObjective.constrictTextToSize = false;
_eventObjective.multiline = true;
_eventObjective.x = 55;
_eventObjective.y = 348;

_eventReward = new Label(12, 0xf0f0f0, 956, 100, true, 1);
_eventReward.align = TextFormatAlign.LEFT;
_eventReward.constrictTextToSize = false;
_eventReward.multiline = true;
_eventReward.x = 55;
_eventReward.y = 368;

_eventBuffs = new Label(12, 0xf0f0f0, 956, 100, true, 1);
_eventBuffs.align = TextFormatAlign.LEFT;
_eventBuffs.constrictTextToSize = false;
_eventBuffs.multiline = true;
_eventBuffs.x = 55;
_eventBuffs.y = 388;

_eventDescriptionContainer = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_INNER,
PanelEnum.HEADER_NOTCHED, 480, 216, 38, 46, 67);

_eventImageContainer = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_eventImageContainer.width = 452;
_eventImageContainer.height = 252;
_eventImageContainer.x = 551;
_eventImageContainer.y = 67;

_eventObjectiveContainer = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);
_eventObjectiveContainer.width = 958;
_eventObjectiveContainer.height = 70;
_eventObjectiveContainer.x
```

```
= 46;
_eventObjectiveContainer.y = 343;

_eventRewardsContainer = UIFactory.getScaleBitmap(PanelEnum.CONTAINER_INNER);


_eventImage = ObjectPool.get(ImageComponent);
_eventImage.init(450, 250);
_eventImage.x = _eventImageContainer.x + 1;
_eventImage.y = _eventImageContainer.y + 1;


_eventProgressBar = UIFactory.getProgressBar(UIFactory.getPanel(PanelEnum.STATBAR,
893, 16), UIFactory.getPanel(PanelEnum.STATBAR_CONTAINER, 901, 24), 0, 1, 0,
_eventObjectiveContainer.
x,
_eventObjectiveContainer.
y + _eventObjectiveContainer.height + 24);

_score = new Label(14, 0xf0f0f0, _eventProgressBar.width, _eventProgressBar.height, true, 1);
_score.align = TextFormatAlign.CENTER;
_score.constrictTextToSize = false;
_score.multiline = true;
_score.x = _eventProgressBar.x;
_score.y = _eventProgressBar.y;



_eventRewards = new Sprite();
_eventRewards.y = _eventProgressBar.y + _eventProgressBar.height + 24;

_nextReward = new EventReward(50, 50, true, true, _tooltips, presenter);
_nextReward.x = _eventProgressBar.x + _eventProgressBar.width + 1;
_nextReward.y = _eventProgressBar.y - 21;
_nextReward.active = true;

_nextArrow = UIFactory.getBitmap('NextArrowUnlitBMD');
_nextArrow.x = _nextReward.x - 5;
_nextArrow.y = _eventProgressBar.y + 4;

addChild(_bg);
addChild(_eventDescriptionContainer);
addChild(_eventObjectiveContainer);
addChild(_eventImageContainer);
addChild(_eventRewardsContainer);
addChild(_eventTitle);
addChild(_eventDescription);
addChild(_eventObjective);
addChild(_eventReward);
addChild(_eventBuffs);
addChild(_eventProgressBar);
```

```
addChild(_score);
addChild(_eventImage);
addChild(_eventRewards);
addChild(_nextReward);
addChild(_nextArrow);

_currentActiveEvent = presenter.currentActiveEvent;

presenter.onAddAchievementsUpdatedListener(onAchievementsUpdated);
presenter.requestAchievements();

setUp();

addEffects();
effectsIN();
}

private function setUp():void
{
if (_currentActiveEvent)
{
var asset:AssetVO = presenter.getAssetVO(_currentActiveEvent.prototype);
_eventTitle.text = asset.visibleName;
_eventDescription.text = asset.descriptionText;
_eventObjective.text = _currentActiveEvent.objectiveText;
_eventReward.text = _currentActiveEvent.rewardsText;
_eventBuffs.text = _currentActiveEvent.activeEventBuffsText;

presenter.loadIcon(asset.largeImage, _eventImage.onImageLoaded);

updateEvent(0);
}
}

private function onAchievementsUpdated( achievements:Dictionary, scores:Dictionary ):void
{
if (_currentActiveEvent && scores.hasOwnProperty(_currentActiveEvent.scoreKey))
updateEvent(scores[_currentActiveEvent.scoreKey].value);
}

private function updateEvent( currentScore:uint ):void
{
if (_activeArrow)
TweenLite.killTweensOf(_activeArrow);

_activeArrow = null;

_nextReward.reward = null;
_nextReward.index = 0;
_eventRewards.removeChildren(0);
```

```
var currentReward:Object;
var currentEventReward:EventReward;
var previousEventReward:EventReward;
var nextArrow:Sprite;
var previousArrow:Sprite;
var rewardPrototype:IPrototype;
var unlocked:Boolean;
var index:uint;
var nextRewardSet:Boolean;
var xPos:Number;
var padding:Number = 26;
var rewards:Array = new Array();
var len:uint = _currentActiveEvent.rewards.length;
var i:uint;
for (; i < len; ++i)
{
currentReward = _currentActiveEvent.rewards[i];
if (!currentReward.hasOwnProperty("factionRequirement") ||
(currentReward.hasOwnProperty("factionRequirement") && currentReward.factionRequirement
== CurrentUser.faction))
rewards.push(currentReward);
}

len = rewards.length;

for (i = 0; i < len; ++i)
{
currentReward = rewards[i];

previousEventReward = currentEventReward;
previousArrow = nextArrow;

rewardPrototype = presenter.getResearchItemPrototypeByName(currentReward.blueprint);
unlocked = currentScore >= currentReward.scoreRequirement;

currentEventReward = new EventReward(100, 100, false, unlocked, _tooltips, presenter);
currentEventReward.reward = rewardPrototype;
currentEventReward.index = index;
currentEventReward.scoreRequirement = currentReward.scoreRequirement;
currentEventReward.x = xPos;
_eventRewards.addChildAt(currentEventReward, index);
++index;

if (previousArrow)
{
var scoreToUse:int = (currentScore > currentReward.scoreRequirement) ?
currentReward.scoreRequirement : currentScore;
_tooltips.addTooltip(previousArrow, this, null, "Current Score: " + scoreToUse + "\nRequired
Score:
```

```
" +
currentReward.scoreRequirement);
}

if (!unlocked && !nextRewardSet)
{
nextRewardSet = true;
_activeArrow = previousArrow;
currentEventReward.active = true;

_nextReward.reward = rewardPrototype;
_nextReward.index = index;

if (previousEventReward)
var previousScoreRequirement:uint = previousEventReward.scoreRequirement;

_score.text = (currentScore - previousScoreRequirement) + '/' +
(currentEventReward.scoreRequirement - previousScoreRequirement);
var percent:Number = (currentScore - previousScoreRequirement) /
(currentEventReward.scoreRequirement - previousScoreRequirement);
_eventProgressBar.amount = percent;

onFadeIn();
} else if (i == (len - 1) && unlocked)
{
_nextArrow.visible = false;
_score.visible = false;
_eventProgressBar.visible = false;
}

if (i != (len - 1))
{
nextArrow = new Sprite();
nextArrow.addChild(UIFactory.getBitmap((unlocked) ? "NextArrowLargeBMD" :
'NextArrowLargeUnlitBMD'));
nextArrow.x = currentEventReward.x + currentEventReward.width;
nextArrow.y = currentEventReward.y + (currentEventReward.height - nextArrow.height) * 0.5 +
9;
_eventRewards.addChild(nextArrow);
}

xPos = currentEventReward.x + currentEventReward.width + padding;
}

_eventRewardsContainer.width = len * 100 + (len - 1) * padding + 20;
_eventRewardsContainer.height = 135;


_eventRewardsContainer.x = _bg.x + (_bg.width - _eventRewardsContainer.width) * 0.5 + 8;
_eventRewardsContainer.y = _eventRewards.y;

_eventRewards.x
```

```
= _eventRewardsContainer.x + 1;
}

private function onFadeOut():void
{
if (_activeArrow)
TweenLite.to(_activeArrow, 0.75, {alpha:1, onComplete:onFadeIn});
}

private function onFadeIn():void
{
if (_activeArrow)
TweenLite.to(_activeArrow, 0.75, {alpha:0.7, onComplete:onFadeOut});
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set tooltips( v:Tooltips ):void { _tooltips = v; }

[Inject]
public function set presenter( value:IEventPresenter ):void { _presenter = value; }
public function get presenter():IEventPresenter { return IEventPresenter(_presenter); }

override public function destroy():void
{
presenter.onRemoveAchievementsUpdatedListener(onAchievementsUpdated);
super.destroy();

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_activeArrow)
TweenLite.killTweensOf(_activeArrow);

_activeArrow = null;

if (_nextReward)
{
TweenLite.killTweensOf(_nextReward);
_nextReward.destroy();
}

_nextReward = null;

if (_eventProgressBar)
ObjectPool.give(_eventProgressBar);

_eventProgressBar
```

```
= null;

if (_eventTitle)
_eventTitle.destroy()

_eventTitle = null;

if (_eventDescription)
_eventDescription.destroy()

_eventDescription = null;

if (_eventObjective)
_eventObjective.destroy()

_eventObjective = null;

if (_eventReward)
_eventReward.destroy()

_eventReward = null;

if (_eventBuffs)
_eventBuffs.destroy()

_eventBuffs = null;

if (_score)
_score.destroy()

_score = null;

if (_eventImage)
ObjectPool.give(_eventImage);

_eventImage = null;

_eventObjectiveContainer = null;
_eventImageContainer = null;
_eventRewardsContainer = null;
_nextArrow = null;
_eventDescriptionContainer = null;
_eventRewards = null;
_currentActiveEvent = null;
}
}
}

---
File 813: igw\com\ui\modal\fakelogin\FakeLoginView.as
package
```

```
com.ui.modal.fakelogin
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.shared.ObjectPool;

public class FakeLoginView extends View
{
private var _bg:DefaultWindowBG;

private var _idBackground:ScaleBitmap;
private var _playerTokenBackground:ScaleBitmap;

private var _loginID:Label;
private var _loginPlayerToken:Label;

private var _okBtn:BitmapButton;

private var _callback:Function;

[PostConstruct]
override public function init():void
{
super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(563, 150);
_bg.addTitle('Fake Login', 114);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_idBackground = UIFactory.getScaleBitmap(PanelEnum.INPUT_BOX_BLUE);
_idBackground.x = 10;
_idBackground.y = 60;
_idBackground.width = 572;
_idBackground.height = 25;

_playerTokenBackground = UIFactory.getScaleBitmap(PanelEnum.INPUT_BOX_BLUE);
_playerTokenBackground.x = 10;
_playerTokenBackground.y = 100;
_playerTokenBackground.width
```

```
= 572;
_playerTokenBackground.height = 25;

_loginID = new Label(16, 0xf0f0f0, 572, 25);
_loginID.align = TextFormatAlign.CENTER;
_loginID.restrict = '0-9';
_loginID.x = _idBackground.x;
_loginID.y = _idBackground.y;

_loginPlayerToken = new Label(16, 0xf0f0f0, 572, 25);
_loginPlayerToken.align = TextFormatAlign.CENTER;
_loginPlayerToken.restrict = 'A-Za-z0-9_';
_loginPlayerToken.x = _playerTokenBackground.x;
_loginPlayerToken.y = _playerTokenBackground.y;

_okBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 100, 30, 0, 0, 'OK', LabelEnum.H1);
_okBtn.x = _playerTokenBackground.x + (_playerTokenBackground.width - _okBtn.width) * 0.5;
_okBtn.y = _playerTokenBackground.y + _playerTokenBackground.height + 20;
addListener(_okBtn, MouseEvent.CLICK, onMouseClick);

addChild(_bg);
addChild(_idBackground);
addChild(_playerTokenBackground);
addChild(_loginID);
addChild(_loginPlayerToken);
addChild(_okBtn);


addEffects();
effectsIN();
}

private function onMouseClick( e:MouseEvent ):void
{
_callback(_loginID.text, _loginPlayerToken.text);
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

public function set callBack( v:Function ):void
{
_callback = v;
}

override public function destroy():void
{
super.destroy();

_idBackground = null;
_playerTokenBackground
```

```
= null;

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_loginID)
_loginID.destroy();

_loginID = null;

if (_loginPlayerToken)
_loginPlayerToken.destroy();

_loginPlayerToken = null;
}
}
}

-----------------------------------
File 814: igw\com\ui\modal\fullscreenprompt\FullScreenPromptModal.as
package com.ui.modal.fullscreenprompt
{
import com.enum.TypeEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.StarbaseEvent;
import com.model.asset.AssetModel;
import com.presenter.shared.IUIPresenter;
import com.service.loading.LoadPriority;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.traderoute.overview.TradeRouteOverviewView;

import flash.display.DisplayObject;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.text.TextFormat;
import flash.text.TextFormatAlign;
import flash.utils.Timer;

import org.parade.core.IView;
import
```

```
org.shared.ObjectPool;

public class FullScreenPromptModal extends View
{
private var _bg:DefaultWindowBG;
private var _message:Label;

private var _yesBtn:BitmapButton;
private var _noBtn:BitmapButton;

private var _titleString:String = "CodeString.FullscreenView.Body"; //FULLSCREEN
private var _messageString:String = "CodeString.FullscreenView.Title"; //Imperium is best
played in fullscreen.<BR><BR>Would you like to fullscreen?

private var _yesBtnString:String = "CodeString.Shared.YesBtn"; //YES
private var _noBtnText:String = 'CodeString.Shared.NoBtn'; //NO

[PostConstruct]
override public function init():void
{
super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.addTitle(_titleString, 180);
_bg.setBGSize(435, 220);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_message = new Label(24, 0xf0f0f0, 400, 140);
_message.constrictTextToSize = false;
_message.multiline = true;
_message.x = 25;
_message.y = 65;
_message.htmlText = _messageString;

_yesBtn = UIFactory.getButton(ButtonEnum.GREEN_A, 180, 40, 20, _bg.height + 8,
_yesBtnString);
addListener(_yesBtn, MouseEvent.CLICK, onOkButtonClick);

_noBtn = UIFactory.getButton(ButtonEnum.RED_A, 180, 40, _yesBtn.x + _yesBtn.width + 60,
_bg.height + 8, _noBtnText);
addListener(_noBtn, MouseEvent.CLICK, onCancelButtonClick);

addChild(_bg);
addChild(_message);
addChild(_yesBtn);
addChild(_noBtn);

addEffects();
effectsIN();
}

private
```

```
function onOkButtonClick( e:MouseEvent ):void
{
presenter.toggleFullScreen();
destroy();
}

private function onCancelButtonClick( e:MouseEvent ):void
{
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
presenter.dispatch(new StarbaseEvent(StarbaseEvent.WELCOME_BACK));
super.destroy();

if (_bg)
ObjectPool.give(_bg);
_bg = null;

if (_message)
_message.destroy();
_message = null;

if (_yesBtn)
_yesBtn.destroy();
_yesBtn = null;

if (_noBtn)
_noBtn.destroy();
_noBtn = null;
}
}
}

------------------------------------
File 815: igw\com\ui\modal\ignore\IgnoreEntry.as
package com.ui.modal.ignore
{
import com.enum.ui.ButtonEnum;
import com.model.player.PlayerVO;
import com.ui.UIFactory;
import com.ui.core.component.button.BitmapButton;
import
```

```
com.ui.core.component.label.Label;
import com.ui.core.ScaleBitmap;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import org.osflash.signals.Signal;

public class IgnoreEntry extends Sprite
{
public var onUnignoreClicked:Signal;

private var _player:PlayerVO;

private var _bg:ScaleBitmap;

private var _name:Label;

private var _unignoreBtn:BitmapButton;

public function IgnoreEntry( player:PlayerVO )
{
super();

onUnignoreClicked = new Signal(IgnoreEntry);

_player = player;

_bg = UIFactory.getScaleBitmap('BtnEmptyUpBMD');
_bg.scale9Grid = new Rectangle(10, 10, 2, 2);
_bg.width = 300;
_bg.height = 20;

_name = new Label(16, 0xf0f0f0, 213, 39, false);
_name.x = 4;
_name.y = 4;
_name.constrictTextToSize = false;
_name.align = TextFormatAlign.LEFT;
_name.text = _player.name;

_unignoreBtn = UIFactory.getButton(ButtonEnum.CLOSE);
_unignoreBtn.x = 267;
_unignoreBtn.y = _bg.y + (_bg.height - _unignoreBtn.height) * 0.5;
_unignoreBtn.addEventListener(MouseEvent.CLICK, onUnignoreBtnClicked)

addChild(_bg);
addChild(_name);
addChild(_unignoreBtn);
}
```

```
private function onUnignoreBtnClicked( e:MouseEvent ):void { onUnignoreClicked.dispatch(this);
}

public function get playerID():String { return _player.id; }

override public function get height():Number { return _bg.height; }

override public function get width():Number { return _bg.width; }

public function destroy():void
{
if (onUnignoreClicked)
onUnignoreClicked.removeAll();

onUnignoreClicked = null;

if (_name)
_name.destroy();

_name = null;

_bg = null;
}
}
}

-----------------------------------
File 816: igw\com\ui\modal\ignore\IgnoreListView.as
package com.ui.modal.ignore
{
import com.model.player.PlayerVO;
import com.presenter.shared.IChatPresenter;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;

import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public class IgnoreListView extends View
{
private var _bg:DefaultWindowBG;

private var _scrollbar:VScrollbar;
private
```

```
var _maxHeight:int;

private var _scrollRect:Rectangle;

private var _holder:Sprite;

private var _title:String = 'CodeString.IgnoreList.Title'; //IGNORE LIST

[PostConstruct]
override public function init():void
{
super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(335, 235);
_bg.addTitle(_title, 114);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_holder = new Sprite();
_holder.x = 26;
_holder.y = 60;

_scrollRect = new Rectangle(_holder.x, _holder.y, 300, 200);
_scrollRect.y = 0;
_holder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 4, 5, 3);
var scrollbarXPos:Number = _bg.width - 24;
var scrollbarYPos:Number = 61;
_scrollbar.init(7, _scrollRect.height, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 34;

addChild(_bg);
addChild(_holder);
addChild(_scrollbar);

presenter.addOnPlayerVOAddedListener(addPlayer);

setUp();
addEffects();
effectsIN();
}

private function setUp():void
{
var
```

```
blockedUsers:Vector.<String> = presenter.blockedUsers;
for (var i:uint = 0; i < blockedUsers.length; ++i)
{
presenter.requestPlayer(blockedUsers[i]);
}
}

private function addPlayer( v:PlayerVO ):void
{
var entry:IgnoreEntry = new IgnoreEntry(v);
entry.onUnignoreClicked.add(removePlayer);
_holder.addChild(entry);

layout();
}

private function removePlayer( v:IgnoreEntry ):void
{
if (v)
{
presenter.blockOrUnblockPlayer(v.playerID);
_holder.removeChild(v);
v.destroy();
v = null;

layout();
}
}

private function layout():void
{
var entry:IgnoreEntry;
var yPos:int = 0;
var xPos:int = _holder.x;
var len:uint = _holder.numChildren;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
entry = IgnoreEntry(_holder.getChildAt(i));
entry.x = xPos;
entry.y = yPos;

yPos += entry.height + 4;

if (i == (len - 1))
_maxHeight += entry.height;
else
_maxHeight += entry.height + 4;
}
_scrollbar.updateScrollableHeight(_maxHeight);

if
```

```
(_maxHeight <= _scrollRect.height)
_scrollbar.resetScroll();
else
onChangedScroll(_scrollbar.percent);
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_holder.scrollRect = _scrollRect;
}

override public function get width():Number { return _bg.width }
override public function get height():Number { return _bg.width }

[Inject]
public function set presenter( v:IChatPresenter ):void { _presenter = v; }
public function get presenter():IChatPresenter { return IChatPresenter(_presenter); }

override public function destroy():void
{
presenter.removeOnPlayerVOAddedListener(addPlayer);
super.destroy();
_bg = null;
}
}
}

-----------------------------------
File 817: igw\com\ui\modal\information\BaseActionPromptModal.as
package com.ui.modal.information
{
import com.enum.TypeEnum;
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.event.StarbaseEvent;
import com.model.asset.AssetModel;
import com.presenter.shared.IUIPresenter;
import com.service.loading.LoadPriority;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.construction.ConstructionView;
import com.ui.modal.traderoute.overview.TradeRouteOverviewView;

import
```

```
flash.display.DisplayObject;
import flash.events.MouseEvent;
import flash.text.TextFormat;
import flash.text.TextFormatAlign;

import org.parade.core.IView;
import org.shared.ObjectPool;

public class BaseActionPromptModal extends View
{
static public const BUILD_ACTION:uint = 1 << 0;
static public const TRADE_ACTION:uint = 1 << 1;
static public const RESEARCH_WEAPONS_ACTION:uint = 1 << 2;
static public const RESEARCH_DEFENSE_ACTION:uint = 1 << 3;
static public const RESEARCH_TECH_ACTION:uint = 1 << 4;
static public const RESEARCH_SHIPS_ACTION:uint = 1 << 5;

private var _titleString:String = "CodeString.ActionPrompt.Title";
private var _messageString:String = "CodeString.ActionPrompt.Message";

private var _buildBtnString:String = "CodeString.ActionPrompt.BuildButton";
private var _researchBtnString:String = "CodeString.ActionPrompt.ResearchButton";
private var _tradeBtnString:String = "CodeString.ActionPrompt.TradeButton";

public var actionTypes:uint = 0;

[Inject]
public var assetModel:AssetModel;

[PostConstruct]
override public function init():void
{
super.init();

createChildren();
updateDisplayList();

addEffects();
effectsIN();
}

private var _childrenCreated:Boolean;

private var _bg:DefaultWindowBG;
private var _msgLbl:Label;
private var _msgBkgd:ScaleBitmap;

private var _buildImg:ImageComponent;
private var _buildBtn:BitmapButton;

private
```

```
var _researchImg:ImageComponent;
private var _researchBtn:BitmapButton;

private var _tradeImg:ImageComponent;
private var _tradeBtn:BitmapButton;

static private const IMG_ENABLED_SUFFIX:String = "_Image_Active.png";
static private const IMG_DISABLED_SUFFIX:String = "_Image_NonActive.png";

private var _btns:Vector.<DisplayObject> = Vector.<DisplayObject>([]);
private var _imgs:Vector.<DisplayObject> = Vector.<DisplayObject>([]);

protected function createChildren():void
{
if (_childrenCreated)
return;

var imgURL:String;

_bg = ObjectPool.get(DefaultWindowBG);
_bg.addTitle(_titleString, 180);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);
addChild(_bg);

_msgBkgd = UIFactory.getPanel(PanelEnum.CONTAINER_NOTCHED_RIGHT_SMALL, 550,
32);
addChild(_msgBkgd);

_msgLbl = UIFactory.getLabel(LabelEnum.SUBTITLE, 550, 32);
_msgLbl.align = TextFormatAlign.LEFT;
_msgLbl.text = _messageString;
var tf:TextFormat = _msgLbl.defaultTextFormat;
tf.font = "Agency";
_msgLbl.defaultTextFormat = tf;
addChild(_msgLbl);

//BUILD
_buildImg = ObjectPool.get(ImageComponent);
_buildImg.init(188, 214);
_buildImg.center = true;
_buildImg.name = "img_" + BUILD_ACTION.toString();
imgURL = "assets/PromptForAction/Build";
imgURL += BUILD_ACTION == (actionTypes & BUILD_ACTION) ? IMG_ENABLED_SUFFIX :
IMG_DISABLED_SUFFIX;
assetModel.getFromCache(imgURL, _buildImg.onImageLoaded, LoadPriority.MEDIUM);
addChild(_buildImg);
_imgs.push(_buildImg);

_buildBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 188, 40, 0, 0, _buildBtnString);
_buildBtn.name = "btn_" + BUILD_ACTION.toString();
_buildBtn.enabled
```

```
= BUILD_ACTION == (actionTypes & BUILD_ACTION);
_buildBtn.addEventListener(MouseEvent.CLICK, onButtonClick);
addChild(_buildBtn);
_btns.push(_buildBtn);

//RESEARCH
var researchType:uint;

if (RESEARCH_WEAPONS_ACTION == (actionTypes & RESEARCH_WEAPONS_ACTION))
researchType = RESEARCH_WEAPONS_ACTION;

else if (RESEARCH_DEFENSE_ACTION == (actionTypes &
RESEARCH_DEFENSE_ACTION))
researchType = RESEARCH_DEFENSE_ACTION;

else if (RESEARCH_TECH_ACTION == (actionTypes & RESEARCH_TECH_ACTION))
researchType = RESEARCH_TECH_ACTION;

else if (RESEARCH_SHIPS_ACTION == (actionTypes & RESEARCH_SHIPS_ACTION))
researchType = RESEARCH_SHIPS_ACTION;

_researchImg = ObjectPool.get(ImageComponent);
_researchImg.init(188, 214);
_researchImg.center = true;
_researchImg.name = "img_" + researchType.toString();
imgURL = "assets/PromptForAction/Research";
imgURL += researchType > 0 ? IMG_ENABLED_SUFFIX : IMG_DISABLED_SUFFIX;
assetModel.getFromCache(imgURL, _researchImg.onImageLoaded, LoadPriority.MEDIUM);
addChild(_researchImg);
_imgs.push(_researchImg);

_researchBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 188, 40, 0, 0, _researchBtnString);
_researchBtn.name = "btn_" + researchType.toString();
_researchBtn.enabled = researchType > 0;
_researchBtn.addEventListener(MouseEvent.CLICK, onButtonClick);
addChild(_researchBtn);
_btns.push(_researchBtn);

//TRADE
_tradeImg = ObjectPool.get(ImageComponent);
_tradeImg.init(188, 214);
_tradeImg.center = true;
_tradeImg.name = "img_" + TRADE_ACTION.toString();
imgURL = "assets/PromptForAction/TradeRoute";
imgURL += TRADE_ACTION == (actionTypes & TRADE_ACTION) ? IMG_ENABLED_SUFFIX
: IMG_DISABLED_SUFFIX;
assetModel.getFromCache(imgURL, _tradeImg.onImageLoaded, LoadPriority.MEDIUM);
addChild(_tradeImg);
_imgs.push(_tradeImg);

_tradeBtn
```

```
= UIFactory.getButton(ButtonEnum.BLUE_A, 188, 40, 0, 0, _tradeBtnString);
_tradeBtn.name = "btn_" + TRADE_ACTION.toString();
_tradeBtn.enabled = TRADE_ACTION == (actionTypes & TRADE_ACTION);
_tradeBtn.addEventListener(MouseEvent.CLICK, onButtonClick);
addChild(_tradeBtn);
_btns.push(_tradeBtn);

_childrenCreated = true;
}

public function updateDisplayList():void
{
const PADDING_TOP:Number = 60;
const PADDING_BOTTOM:Number = 10;
const PADDING_LEFT:Number = 35;
const PADDING_RIGHT:Number = 10;
const GAP:Number = 15;

var iw:Number = 188 * 3 + GAP * 2;
var tx:Number = PADDING_LEFT;
var tw:Number = PADDING_LEFT + iw + PADDING_RIGHT;

_msgBkgd.x = PADDING_LEFT;
_msgBkgd.y = PADDING_TOP;
_msgBkgd.setSize(iw, 32);

_msgLbl.x = PADDING_LEFT + 3;
_msgLbl.y = PADDING_TOP + 3;
_msgLbl.setSize(iw, 32);

var uic:DisplayObject;
var i:int;

for (i; i < _btns.length; i++)
{
uic = _imgs[i];
uic.x = tx;
uic.y = _msgBkgd.y + _msgBkgd.height + GAP;

uic = _btns[i];
uic.x = tx;
uic.y = _msgBkgd.y + _msgBkgd.height + GAP + 214 + GAP;

tx += uic.width + GAP;
}

var th:Number = _msgBkgd.y + _msgBkgd.height + GAP + 214 + GAP + 40;

// _bg.setBGSize(tw, th);
_bg.setBGSize(635, 360);
}
```

```
private function onButtonClick( event:MouseEvent ):void
{
var name:String = event.target.name;
var id:uint;

if (name && name.indexOf("_") > -1)
{
var ns:Array = name.split("_");
id = uint(ns[1]);
}

if (id == 0)
return;

var view:IView;

switch (id)
{
case BUILD_ACTION:
{
view = presenter.getView(ConstructionView);
if (!view)
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.BUILD, null, null);
} else
view = null;
break;
}

case RESEARCH_WEAPONS_ACTION:
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH,
TypeEnum.WEAPONS_FACILITY, null);
break;
}

case RESEARCH_DEFENSE_ACTION:
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH,
TypeEnum.DEFENSE_DESIGN, null);
break;
}

case RESEARCH_TECH_ACTION:
{
view
```

```
= _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH,
TypeEnum.ADVANCED_TECH, null);
break;
}

case RESEARCH_SHIPS_ACTION:
{
view = _viewFactory.createView(ConstructionView);
ConstructionView(view).openOn(ConstructionView.RESEARCH, TypeEnum.SHIPYARD, null);
break;
}

case TRADE_ACTION:
{
view = _viewFactory.createView(TradeRouteOverviewView);
break;
}
}

if (view)
_viewFactory.notify(view);

destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
presenter.dispatch(new StarbaseEvent(StarbaseEvent.WELCOME_BACK));
super.destroy();

if (_bg)
ObjectPool.give(_bg);
_bg = null;

_msgBkgd = null;

if (_msgLbl)
_msgLbl.destroy();

_msgLbl = null;

if (_buildImg)
ObjectPool.give(_buildImg);
```

```
_buildImg = null;

if (_researchImg)
ObjectPool.give(_researchImg);

_researchImg = null;

if (_tradeImg)
ObjectPool.give(_tradeImg);

_tradeImg = null;

if (_buildBtn)
{
_buildBtn.removeEventListener(MouseEvent.CLICK, onButtonClick);
_buildBtn.destroy();
}

_buildBtn = null;

if (_researchBtn)
{
_researchBtn.removeEventListener(MouseEvent.CLICK, onButtonClick);
_researchBtn.destroy();
}

_researchBtn = null;

if (_tradeBtn)
{
_tradeBtn.removeEventListener(MouseEvent.CLICK, onButtonClick);
_tradeBtn.destroy();
}

_tradeBtn = null;
}
}
}

-----------------------------------
File 818: igw\com\ui\modal\information\DailyRewardView.as
package com.ui.modal.information
{
import com.Application;
import com.enum.ui.PanelEnum;
import com.event.StarbaseEvent;
import com.event.StateEvent;
import com.model.asset.AssetVO;
import
```

```
com.model.blueprint.BlueprintVO;
import com.model.player.CurrentUser;
import com.model.prototype.IPrototype;
import com.model.prototype.PrototypeModel;
import com.presenter.shared.IUIPresenter;
import com.service.language.Localization;
import com.service.server.incoming.starbase.StarbaseDailyRewardResponse;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.adobe.utils.StringUtil;
import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.shared.ObjectPool;

public class DailyRewardView extends View
{
private var _bg:Sprite;
private var _closeBtn:BitmapButton;
private var _windowTitle:Label;
private var _header:Label;
private var _subheader:Label;
private var _iconFrames:Vector.<Bitmap>;
private var _resourceLbls:Vector.<Label>;
private var _rewards:StarbaseDailyRewardResponse;
private var _numRewardIcons:int;
private var _collectRewardBtn:BitmapButton;

private var _blueprintProtoName:String;
private var _blueprintHolder:Sprite;
private var _blueprintFrame:Bitmap;
private var _blueprintShipIcon:ImageComponent;
private var _buffIcon:ImageComponent;
private var _purchaseBlueprintBtn:BitmapButton;
private
```

```
var _blueprintBG:Bitmap;
private var _blueprintNameBG:Bitmap;
private var _blueprintGlow:Bitmap;
private var _blueprintPremiumSymbol:Bitmap;
private var _blueprintName:Label;
private var _bpCollectedNumbers:Label;
private var _blueprintCompleteCost:Label;
private var _blueprint:BlueprintVO;
private var _blueprintCost:int;
private var _buff:IPrototype;
private var _blueprintsCollectedString:String = 'CodeString.Shared.OutOf';
//[[Number.MinValue]]/[[Number.MaxValue]]

private var _uiPresenter:IUIPresenter;

private var _windowTitleString:String = 'CodeString.RewardView.Title';
private var _gratsBroString:String = 'CodeString.RewardView.GratsBro';
private var _subtitleString:String = 'CodeString.RewardView.Subtitle';

private var _tooltips:Tooltips;

[PostConstruct]
override public function init():void
{
super.init();

_closeBtn = ButtonFactory.getCloseButton(490, 19);
addListener(_closeBtn, MouseEvent.CLICK, onCollectReward);

_resourceLbls = new Vector.<Label>;
_iconFrames = new Vector.<Bitmap>;
_numRewardIcons = 0;

var mcBGClass:Class = Class(getDefinitionByName('DailyRewardMC'));
_bg = Sprite(new mcBGClass());
addChild(_bg);


_collectRewardBtn = ButtonFactory.getBitmapButton('BtnRewardUpBMD', _bg.x + _bg.width -
250, _bg.y + _bg.height + 2, 'COLLECT REWARDS', 0xacd1ff, 'BtnRewardROBMD',
'BtnRewardDownBMD');
_collectRewardBtn.fontSize = 28;
_collectRewardBtn.addEventListener(MouseEvent.CLICK, onCollectReward);
addChild(_collectRewardBtn);


_windowTitle = new Label(30, 0xd1e5f7, 383, 5, true);
_windowTitle.x = 30;
_windowTitle.y = 5;
_windowTitle.constrictTextToSize = false;
_windowTitle.align
```

```
= TextFormatAlign.LEFT;
_windowTitle.autoSize = TextFieldAutoSize.LEFT;
_windowTitle.text = _windowTitleString;
addChild(_windowTitle);

_header = new Label(44, 0xd1e5f7, 518, 5, true);
addChild(_header);

_subheader = new Label(20, 0xffd785, 518, 5, true);
addChild(_subheader);

if (_rewards.creditsReward > 0)
_numRewardIcons++;

if (_rewards.alloyReward > 0)
_numRewardIcons++;

if (_rewards.energyReward > 0)
_numRewardIcons++;

if (_rewards.syntheticReward > 0)
_numRewardIcons++;

if (_rewards.buffPrototype != '')
_numRewardIcons++;

_buff = _uiPresenter.getBuffPrototypeByName(_rewards.buffPrototype);

_buffIcon = ObjectPool.get(ImageComponent);
_buffIcon.init(50, 50);


_blueprintProtoName = _rewards.blueprintPrototype; // = 'Fighter_IGA_Legendary';
if (_blueprintProtoName)
showBlueprint();

layout();

addEffects();
effectsIN();
}

private function layout():void
{

_header.x = 21;
_header.constrictTextToSize = false;
_header.align = TextFormatAlign.CENTER;
_header.autoSize = TextFieldAutoSize.CENTER;
_header.text = _gratsBroString;

//determine
```

```
x, y pos based on number of rewards and blueprints gained
var xPos:int = 0;
var yPos:int = 0;
if (_numRewardIcons == 4)
xPos = 137;
else
xPos = 99;

if (_blueprintProtoName)
{
_header.y = 42;
yPos = 233;
} else
{
_header.y = 100;
yPos = 187;
}

_subheader.x = 21;
_subheader.y = _header.y + _header.textHeight + 2;
_subheader.constrictTextToSize = false;
_subheader.align = TextFormatAlign.CENTER;
_subheader.autoSize = TextFieldAutoSize.CENTER;
_subheader.text = _subtitleString;

var tooltip:String;
var loc:Localization = Localization.instance;
for (var i:int = 0; i < _numRewardIcons; i++)
{
_iconFrames[i] = addScaleBitmap(PanelEnum.BLUE_FRAME, xPos + (i * 76), yPos);

_resourceLbls[i] = new Label(18, 0xd1e5f7, 39, 17, false);
_resourceLbls[i].constrictTextToSize = false;

switch (i)
{
case 0:
addBitmap('DailyCreditsIconBMD', _iconFrames[i].x + 5, _iconFrames[i].y + 5);
_resourceLbls[i].text = StringUtil.abbreviateNumber(_rewards.creditsReward);
break;
case 1:
addBitmap('DailyAlloyIconBMD', _iconFrames[i].x + 5, _iconFrames[i].y + 5);
_resourceLbls[i].text = StringUtil.abbreviateNumber(_rewards.alloyReward);
break;
case 2:
addBitmap('DailyEnergyIconBMD', _iconFrames[i].x + 5, _iconFrames[i].y + 5);
_resourceLbls[i].text = StringUtil.abbreviateNumber(_rewards.energyReward);
break;
case 3:
addBitmap('DailySynthIconBMD', _iconFrames[i].x + 5, _iconFrames[i].y + 5);
_resourceLbls[i].text
```

```
= StringUtil.abbreviateNumber(_rewards.syntheticReward);
break;
case 4:
if (_buff)
{
var assetVO:AssetVO = _uiPresenter.getAssetVOFromIPrototype(_buff);
_uiPresenter.loadIcon(assetVO.smallImage, _buffIcon.onImageLoaded);
_buffIcon.x = _iconFrames[i].x + 5;
_buffIcon.y = _iconFrames[i].y + 5;
addChild(_buffIcon);
tooltip = 'Buff\n' + loc.getString(assetVO.visibleName) + '\n' +
loc.getString(assetVO.descriptionText);
_tooltips.addTooltip(_buffIcon, this, null, tooltip);
} else
addBitmap('DailyBuffShieldIconBMD', _iconFrames[i].x + 5, _iconFrames[i].y + 5);
break;
}

if (i != 4)
addBitmap('TextBackingBMD', _iconFrames[i].x + 17, _iconFrames[i].y + 39);

_resourceLbls[i].x = _iconFrames[i].x + 18;
_resourceLbls[i].y = _iconFrames[i].y + 37;

addChild(_resourceLbls[i]);
}

}

private function addBitmap( className:String, dx:Number, dy:Number ):Bitmap
{
var bmpClass:Class = Class(getDefinitionByName((className)));
var newBmp:Bitmap;
newBmp = new Bitmap(BitmapData(new bmpClass));
newBmp.x = dx;
newBmp.y = dy;

addChild(newBmp);

return newBmp;
}

private function addScaleBitmap( className:String, dx:Number, dy:Number ):Bitmap
{
var newBmp:ScaleBitmap = UIFactory.getScaleBitmap(className);
newBmp.width = 60;
newBmp.height = 60;
newBmp.x = dx;
newBmp.y = dy;

addChild(newBmp);
```

```
return newBmp;
}

private function showBlueprint():void
{
var blueprintBgClass:Class = Class(getDefinitionByName(('LootedBlueprintBMD')));
_blueprintBG = new Bitmap(BitmapData(new blueprintBgClass()));
_blueprintBG.x = 476;
_blueprintBG.y = 342;

var blueprintBgFrameClass:Class = Class(getDefinitionByName('SelectionFrameBMD'));
_blueprintFrame = new Bitmap(BitmapData(new blueprintBgFrameClass()));
_blueprintFrame.x = 136; //485;
_blueprintFrame.y = 121; //331;

_blueprintShipIcon = new ImageComponent()
_blueprintShipIcon.init(100, 100);

_blueprintGlow = PanelFactory.getPanel('BlueprintGlowBMD');
_blueprintGlow.x = 120; //469;
_blueprintGlow.y = 104; //314;
_blueprintGlow.alpha = 0;

_blueprintNameBG = PanelFactory.getPanel('BlueprintPlacardBMD');

_purchaseBlueprintBtn = ButtonFactory.getBitmapButton('SquareBuyBtnNeutralBMD', 293, 127,
'COMPLETE', 0xf7c78b, 'SquareBuyBtnRollOverBMD', 'SquareBuyBtnSelectedBMD');
_purchaseBlueprintBtn.label.fontSize = 22;
_purchaseBlueprintBtn.addEventListener(MouseEvent.CLICK, onBlueprintPurchase, false, 0,
true);

_blueprintCompleteCost = new Label(18, 0xf0f0f0, 114, 25, false);
_blueprintCompleteCost.align = TextFormatAlign.CENTER;
_blueprintCompleteCost.constrictTextToSize = false;
_blueprintCompleteCost.x = _purchaseBlueprintBtn.x + 4;
_blueprintCompleteCost.y = _purchaseBlueprintBtn.y + 26;

_blueprintPremiumSymbol = PanelFactory.getPanel('KalganSymbolBMD');
_blueprintPremiumSymbol.x = _purchaseBlueprintBtn.x + 7;
_blueprintPremiumSymbol.y = _purchaseBlueprintBtn.y + 24;

var blueprintVO:IPrototype =
PrototypeModel.instance.getBlueprintPrototype(_blueprintProtoName);
var bpAsset:AssetVO = _uiPresenter.getAssetVOFromIPrototype(blueprintVO);
//Only ships have schematic images
if (bpAsset.largeImage == '')
_uiPresenter.loadIcon(bpAsset.mediumImage, onBlueprintIconLoaded);
else
_uiPresenter.loadIcon(bpAsset.largeImage,
```

```
onBlueprintIconLoaded);

var rarity:String = blueprintVO.getValue('rarity');
var bpLabelColor:uint = CommonFunctionUtil.getRarityColor(rarity);
_blueprintFrame.filters = [CommonFunctionUtil.getRarityGlow(rarity)];


// Blueprint Name
_blueprintName = new Label(16, bpLabelColor, 176, 26);
_blueprintName.constrictTextToSize = false;
_blueprintName.align = TextFormatAlign.CENTER;
_blueprintName.text = bpAsset.visibleName;
_blueprintName.letterSpacing = 1.5;

var numCollected:int = 0;
_blueprint = _uiPresenter.getBlueprintByName(_blueprintProtoName);
if (_blueprint)
numCollected = _blueprint.partsCollected;
else
++numCollected;

// Blueprints Collected
_bpCollectedNumbers = new Label(12, 0xF0F0F0, 10, 25, true, 1);
_bpCollectedNumbers.constrictTextToSize = false;
_bpCollectedNumbers.autoSize = TextFieldAutoSize.LEFT;
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':numCollected, '[[Number.MaxValue]]':blueprintVO.getValue('parts')});
_bpCollectedNumbers.letterSpacing = 1;

if (_blueprint)
{
if (_blueprint.complete)
{
_purchaseBlueprintBtn.visible = false;
_blueprintCompleteCost.visible = false;
_blueprintPremiumSymbol.visible = false;
_blueprintNameBG.x = 265;
_blueprintNameBG.y = _blueprintFrame.y + 33;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;

onBlueprintGlowFadeOut();
} else
{
_purchaseBlueprintBtn.visible = true;
_blueprintCompleteCost.visible = true;
_blueprintPremiumSymbol.visible
```

```
= true;
_blueprintCost = _uiPresenter.getBlueprintHardCurrencyCost(_blueprint,
_blueprint.partsRemaining);
_blueprintCompleteCost.text = String(_blueprintCost);

_blueprintNameBG.x = 265;
_blueprintNameBG.y = 194;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;
}
} else
{
_blueprintNameBG.x = 265;
_blueprintNameBG.y = 194;

_blueprintName.x = _blueprintNameBG.x + 4;
_blueprintName.y = _blueprintNameBG.y + 3;

_bpCollectedNumbers.x = _blueprintNameBG.x + _blueprintNameBG.width + 1;
_bpCollectedNumbers.y = _blueprintNameBG.y + 3;
}

addChild(_blueprintFrame);
addChild(_blueprintShipIcon);
addChild(_blueprintNameBG);
addChild(_blueprintGlow);
addChild(_blueprintName);
addChild(_bpCollectedNumbers);
addChild(_purchaseBlueprintBtn);
addChild(_blueprintCompleteCost);
addChild(_blueprintPremiumSymbol);
}

private function onBlueprintIconLoaded( asset:BitmapData ):void
{
if (asset && _blueprintShipIcon)
{
_blueprintShipIcon.clearBitmap();
_blueprintShipIcon.onImageLoaded(asset);
_blueprintShipIcon.x = (_blueprintFrame.x + _blueprintFrame.width * 0.5) -
(_blueprintShipIcon.width * 0.5);
_blueprintShipIcon.y = (_blueprintFrame.y + _blueprintFrame.height * 0.5) -
(_blueprintShipIcon.height * 0.5);
}
}

private
```

```
function onBlueprintGlowFadeOut():void
{
TweenLite.to(_blueprintGlow, 1.0, {alpha:1.0, ease:Quad.easeOut,
onComplete:onBlueprintGlowFadeIn, overwrite:0});
}

private function onBlueprintGlowFadeIn():void
{
TweenLite.to(_blueprintGlow, 1.0, {alpha:0.0, ease:Quad.easeIn,
onComplete:onBlueprintGlowFadeOut, overwrite:0});
}

private function onBlueprintPurchase( e:MouseEvent ):void
{
if (_blueprint && !_blueprint.complete)
{
if (CurrentUser.wallet.premium >= _blueprintCost)
{
_uiPresenter.purchaseBlueprint(_blueprint, _blueprint.partsRemaining);
_purchaseBlueprintBtn.visible = false;
_blueprintCompleteCost.visible = false;
_blueprintPremiumSymbol.visible = false;
if (_bpCollectedNumbers)
_bpCollectedNumbers.setTextWithTokens(_blueprintsCollectedString,
{'[[Number.MinValue]]':_blueprint.totalParts, '[[Number.MaxValue]]':_blueprint.totalParts});
onBlueprintGlowFadeOut();

} else
CommonFunctionUtil.popPaywall();
}
}

private function onCollectReward( e:MouseEvent ):void
{
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

public function get rewards():StarbaseDailyRewardResponse { return _rewards; }
public function set rewards( value:StarbaseDailyRewardResponse ):void { _rewards = value; }

[Inject]
public function set presenter( value:IUIPresenter ):void { _uiPresenter = value; }

[Inject]
public function set tooltips( value:Tooltips ):void { _tooltips = value; }

override public function destroy():void
{
```

```
if (Application.STATE == StateEvent.GAME_STARBASE)
_uiPresenter.dispatch(new StarbaseEvent(StarbaseEvent.WELCOME_BACK));

super.destroy();
_bg = null;
_closeBtn.destroy();
_closeBtn = null;

_windowTitle.destroy();
_windowTitle = null;

_header.destroy();
_header = null;

_subheader.destroy();
_subheader = null;

_iconFrames = null;
_resourceLbls = null;

_rewards.destroy();
_rewards = null;

_collectRewardBtn.destroy();
_collectRewardBtn = null;

if (_buffIcon)
ObjectPool.give(_buffIcon);

_buffIcon = null;

_blueprintFrame = null;
_blueprintBG = null;
_blueprintPremiumSymbol = null;

if (_blueprintGlow)
TweenLite.killTweensOf(_blueprintGlow);

_blueprintGlow = null;

if (_blueprintShipIcon)
ObjectPool.give(_blueprintShipIcon);

_blueprintShipIcon = null;

if (_purchaseBlueprintBtn)
_purchaseBlueprintBtn.destroy();

_purchaseBlueprintBtn = null;

if
```

```
(_blueprintName)
_blueprintName.destroy();

_blueprintName = null;

if (_bpCollectedNumbers)
_bpCollectedNumbers.destroy();

_bpCollectedNumbers = null;


if (_blueprintCompleteCost)
_blueprintCompleteCost.destroy();

_blueprintCompleteCost = null;

if (_tooltips)
_tooltips.removeTooltip(null, this);

_tooltips = null;

}
}
}

-----------------------------------
File 819: igw\com\ui\modal\information\GuestRestrictionView.as
package com.ui.modal.information
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;

import com.model.player.CurrentUser;
import com.ui.core.ButtonPrototype;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.label.Label;
import com.ui.core.component.button.BitmapButton;

import flash.display.Sprite;
import flash.display.Stage;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;

import com.service.ExternalInterfaceAPI;

import
```

```
org.shared.ObjectPool;

public class GuestRestrictionView extends View
{
private var _bg:DefaultWindowBG;
private var _info:Label;

private var _holder:Sprite;

private var _titleText:String = 'CodeString.GuestRestriction.Title';

private var _infoText:String = 'CodeString.GuestRestriction.Info';

private var _continueBtn:BitmapButton;
private var _registerBtn:BitmapButton;
private var _continueText:String = 'CodeString.GuestRestriction.ContinueBtn'; //Continue
private var _registerText:String = 'CodeString.GuestRestriction.RegisterBtn'; //Register

[PostConstruct]
override public function init():void
{
super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(563, 200);
_bg.addTitle(_titleText, 114);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_info = new Label(20, 0xf0f0f0, 572, 25);
_info.align = TextFormatAlign.CENTER;
_info.setTextWithTokens(_infoText,null);
_info.y = 90;

_holder = new Sprite();
_holder.x = 25;
_holder.y = 53;

_continueBtn = UIFactory.getButton(ButtonEnum.BLUE_A, 160, 40, 100, 170, _continueText,
LabelEnum.H1);
addListener(_continueBtn, MouseEvent.CLICK, onContinue);
_registerBtn = UIFactory.getButton(ButtonEnum.GREEN_A, 160, 40, 315, 170, _registerText,
LabelEnum.H1);
addListener(_registerBtn, MouseEvent.CLICK, onRegister);


addChild(_bg);
addChild(_holder);
addChild(_info);
addChild(_continueBtn);
addChild(_registerBtn);

addEffects();
```

```
effectsIN();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

private function onContinue( e:MouseEvent ):void
{
onClose();
}
private function onRegister( e:MouseEvent ):void
{
ExternalInterfaceAPI.registerGuest();
}

override public function destroy():void
{
super.destroy();

if (_bg)
ObjectPool.give(_bg);

_bg = null;

if (_info)
_info.destroy();

_info = null;

_holder = null;


if (_continueBtn)
_continueBtn.destroy();
_continueBtn = null;


if (_registerBtn)
_registerBtn.destroy();
_registerBtn = null;
}
}
}

-----------------------------------
File 820: igw\com\ui\modal\information\MessageOfTheDayView.as
package com.ui.modal.information
{

import
```

```
com.event.StarbaseEvent;
import com.event.StateEvent;
import com.model.motd.MotDDailyRewardModel;
import com.model.motd.MotDVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.IUIPresenter;
import com.service.server.incoming.starbase.StarbaseDailyRewardResponse;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.misc.ImageComponent;
import com.ui.core.component.pips.PipComponent;
import com.ui.core.component.pips.PipEvent;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.events.TimerEvent;
import flash.geom.Rectangle;
import flash.text.StyleSheet;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.Timer;
import flash.utils.getDefinitionByName;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.shared.ObjectPool;

public class MessageOfTheDayView extends View
{
public static const MAX_MESSAGES:int = 7;
private static const MAX_DAILY_REWARDS:int = 7;
private static var _isClaimed:Boolean = false;

private var _bg:Sprite;
private var _closeBtn:BitmapButton;
private var _image:ImageComponent;
private var _messageBG:Bitmap;
private var _currentDayImage:Bitmap;
private var _currentChestGlow:Bitmap;
private var _dailyBtn:BitmapButton;
private var _dailyIcons:Vector.<Bitmap>;
private var _dailyCheckbox:Vector.<Bitmap>;

private
```

```
var _message:Label;
private var _windowTitle:Label;
private var _messageTitle:Label;
private var _dailyHeader:Label;
private var _dailySideMessage:Label;

private var _dailyLbls:Vector.<Label>;

private var _scrollbar:VScrollbar;
private var _scrollRect:Rectangle;

private var _messageVOs:Vector.<MotDVO>;

private var _pipComponent:PipComponent;

private var _dailyRewardModel:MotDDailyRewardModel;
private var _loginBonusProto:IPrototype;

private var _claimDay:int;
private var _claimTimer:Timer;

private var _closeWindowBtn:BitmapButton;

private var _windowTitleString:String = 'CodeString.MotD.Title';
private var _DailyRewardString:String = 'CodeString.MotD.RewardTitle'; //DAILY LOGIN
REWARD
private var _DailySideMessageString:String = 'CodeString.MotD.RewardSideTitle'; //Play
Everday for Better Rewards!
private var _DailyClaimedString:String = 'CodeString.MotD.Claimed'; //CLAIMED
private var _closeWindowBtnText:String = 'CodeString.MotD.CloseWindow'; // CLOSE
WINDOW
private var _dayText:String = 'CodeString.MotD.Day'; // DAY [[Number.Day]]
private var _todayText:String = 'CodeString.MotD.Today'; // TODAY

[PostConstruct]
override public function init():void
{
super.init();

_messageVOs = presenter.motdModel.motd;
_dailyRewardModel = presenter.motdDailyModel;

if (_messageVOs.length == 0)
return;

_dailyLbls = new Vector.<Label>;
_dailyIcons = new Vector.<Bitmap>;
_dailyCheckbox = new Vector.<Bitmap>;

if (_dailyRewardModel.canClaimDelta <= 0)
_claimDay
```

```
= _dailyRewardModel.escalation;
else
{
_claimDay = _dailyRewardModel.escalation > 0 ? _dailyRewardModel.escalation - 1 : 6;
_isClaimed = true;
_claimTimer = new Timer(_dailyRewardModel.timeRemainingMS, 1);
addListener(_claimTimer, TimerEvent.TIMER_COMPLETE, update);
_claimTimer.start();
}

presenter.addDailyRewardListener(showRewardView);

var mcBGClass:Class = Class(getDefinitionByName('MotDMC'));
_bg = Sprite(new mcBGClass());

_pipComponent = new PipComponent();
_pipComponent.init(true, true);
_pipComponent.totalPips = _messageVOs.length <= MAX_MESSAGES ? _messageVOs.length
: MAX_MESSAGES;
_pipComponent.x = 243;
_pipComponent.y = 253;
_pipComponent.selected = 0;

addListener(_pipComponent, PipEvent.PIP_CLICKED, onPipClicked);

var i:int;
if (_messageVOs.length > 1)
{
_pipComponent.visible = true;
for (i = 0; i < _messageVOs.length; i++)
_pipComponent.setPipState(i, _messageVOs[i].isRead);
} else
{
_pipComponent.visible = false;
}

_closeBtn = ButtonFactory.getCloseButton(560, 19);
addListener(_closeBtn, MouseEvent.CLICK, onClose);

var messageBG:Class = Class(getDefinitionByName(('MotDTextBGBMD')));
var bgRect:Rectangle = new Rectangle(0, 50, 569, 10);
_messageBG = new ScaleBitmap(BitmapData(new messageBG));
_messageBG.scale9Grid = bgRect;
_messageBG.x = 26;
_messageBG.y = 165;
// addChild(_messageBG);

_windowTitle = new Label(30, 0xd1e5f7, 383, 5, true);
_windowTitle.x = 34;
_windowTitle.y = 5; //14;
_windowTitle.constrictTextToSize
```

```
= false;
_windowTitle.align = TextFormatAlign.LEFT;
_windowTitle.autoSize = TextFieldAutoSize.LEFT;
_windowTitle.text = _windowTitleString;
_windowTitle.letterSpacing = 1.5;

_messageTitle = new Label(22, 0xd1e5f7, 383, 29, true);
_messageTitle.x = 34;
_messageTitle.y = 47;
_messageTitle.constrictTextToSize = false;
_messageTitle.align = TextFormatAlign.LEFT;
_messageTitle.text = _messageVOs[0].title;
_messageTitle.letterSpacing = 1.5;

_message = new Label(12, 0xd1e5f7, 532, 500, true, 1);
_message.x = 32;
_message.y = 210;
_message.constrictTextToSize = false;
_message.multiline = true;
_message.autoSize = TextFieldAutoSize.LEFT;
_message.align = TextFormatAlign.LEFT;
_message.htmlText = _messageVOs[0].message;
_message.letterSpacing = 1.5;
_message.mouseEnabled = true;

var style:StyleSheet = new StyleSheet();
var hover:Object = new Object();
hover.fontWeight = "bold";
hover.color = "#4CACF0";
var link:Object = new Object();
link.fontWeight = "bold";
link.textDecoration = "underline";
link.color = "#4CACF0";
var active:Object = new Object();
active.fontWeight = "bold";
active.color = "#4CACF0";
var visited:Object = new Object();
visited.fontWeight = "bold";
visited.color = "#4CACF0";
visited.textDecoration = "underline";

style.setStyle("a:link", link);
style.setStyle("a:hover", hover);
style.setStyle("a:active", active);
style.setStyle(".visited", visited);

_message.styleSheet = style;

_messageBG.height = _message.textHeight + 20 < 217 ? _message.textHeight + 20 : 217;
_scrollRect = new Rectangle(0, 0, _messageBG.width - 37, _messageBG.height - 15);
_message.scrollRect
```

```
= _scrollRect;

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 5, 5, 2);
var scrollbarXPos:Number = _messageBG.width - 1;
var scrollbarYPos:Number = _messageBG.y + 10;
_scrollbar.init(7, _scrollRect.height - 10, scrollbarXPos, scrollbarYPos, dragBarBGRect, '',
'ScrollbarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.updateScrollableHeight(_message.textHeight);
_scrollbar.maxScroll = 16.3;

_dailyHeader = new Label(22, 0xd1e5f7, 30, 29, true);
_dailyHeader.x = 32;
_dailyHeader.y = 284;
_dailyHeader.constrictTextToSize = false;
_dailyHeader.align = TextFormatAlign.LEFT;
_dailyHeader.autoSize = TextFieldAutoSize.LEFT;
_dailyHeader.text = _DailyRewardString;
_dailyHeader.letterSpacing = 1.5;

_dailySideMessage = new Label(22, 0xffcf4f, 30, 29, true);
_dailySideMessage.x = _dailyHeader.x + _dailyHeader.textWidth + 15;
_dailySideMessage.y = _dailyHeader.y;
_dailySideMessage.constrictTextToSize = false;
_dailySideMessage.align = TextFormatAlign.LEFT;
_dailySideMessage.autoSize = TextFieldAutoSize.LEFT;
_dailySideMessage.text = _DailySideMessageString;
_dailySideMessage.letterSpacing = 1.5;

_image = ObjectPool.get(ImageComponent);
_image.init(579, 120);
_image.x = 27;
_image.y = 81;

_closeWindowBtn = ButtonFactory.getBitmapButton('BtnRewardUpBMD', _bg.x + _bg.width -
250, _bg.y + _bg.height + 2, _closeWindowBtnText, 0xacd1ff, 'BtnRewardROBMD',
'BtnRewardDownBMD');
_closeWindowBtn.fontSize = 28;
addListener(_closeWindowBtn, MouseEvent.CLICK, onClose);

addChild(_bg);
addChild(_closeBtn);
addChild(_pipComponent);
addChild(_windowTitle);
addChild(_messageTitle);
addChild(_message);
addChild(_scrollbar);
addChild(_dailyHeader);
addChild(_dailySideMessage);
```

```
addChild(_image);
addChild(_closeWindowBtn);

setUpRewards();
layoutRewards();
showMessageByIdx(0);

addEffects();
effectsIN();
}

private function setUpRewards():void
{
var chestBitmapName:String;
var checkboxBitmapName:String;
var bitmap:Bitmap;
var alpha:Number;

_currentChestGlow = UIFactory.getBitmap('LoginCrate' + _claimDay + 'GlowBMD');
_currentChestGlow.alpha = 0;
addChild(_currentChestGlow);

for (var i:uint = 0; i < MAX_DAILY_REWARDS; ++i)
{


if (i > _claimDay)
{
chestBitmapName = 'LoginCrate' + i + 'ClosedBMD';
checkboxBitmapName = 'BtnCheckBoxUpBMD';
alpha = 1.0;
} else if (i < _claimDay)
{
chestBitmapName = 'LoginCrate' + i + 'EmptyBMD';
checkboxBitmapName = 'BtnCheckBoxSelectedBMD';
alpha = 0.5;
} else
{
if (!_isClaimed)
{
chestBitmapName = 'LoginCrate' + i + 'ClaimBMD'
checkboxBitmapName = 'BtnCheckBoxUpBMD';
alpha = 1.0;
} else
{
chestBitmapName = 'LoginCrate' + i + 'EmptyBMD';
checkboxBitmapName = 'BtnCheckBoxSelectedBMD';
alpha = 0.5;
}
}
```

```
bitmap = addBitmap(chestBitmapName);
bitmap.alpha = alpha;
bitmap.smoothing = true;

_dailyIcons.push(bitmap);
_dailyCheckbox.push(addBitmap(checkboxBitmapName));
_dailyLbls.push(addDayLabel(i + 1));
}

var btn:BitmapData = new BitmapData(71, 73, true, 0xf0f0f0);
_dailyBtn = new BitmapButton()
_dailyBtn.init(btn, btn, btn, btn, btn);
_dailyBtn.enabled = false;
addChild(_dailyBtn);

if (!_isClaimed)
{
_dailyBtn.enabled = true;
addListener(_dailyBtn, MouseEvent.CLICK, claimReward);
addListener(_dailyBtn, MouseEvent.ROLL_OVER, onChestRollOver);
addListener(_dailyBtn, MouseEvent.ROLL_OUT, onChestRollOut);
onFadeOut(_currentChestGlow);
}

}

private function layoutRewards():void
{
var chestBitmap:Bitmap;
var checkboxBitmap:Bitmap;
var label:Label;

var bitmapXPos:Number = 43;
var labelXPos:Number = 43;
for (var i:uint = 0; i < MAX_DAILY_REWARDS; ++i)
{
chestBitmap = _dailyIcons[i];

chestBitmap.x = bitmapXPos;
if (i != _claimDay || _isClaimed)
{
chestBitmap.y = 398 - chestBitmap.height;
} else
{
chestBitmap.y = 411 - chestBitmap.height;
_currentChestGlow.x = chestBitmap.x;
_currentChestGlow.y = chestBitmap.y;
}

checkboxBitmap
```

```
= _dailyCheckbox[i];
checkboxBitmap.x = chestBitmap.x - 9;
checkboxBitmap.y = 406 - checkboxBitmap.height;

label = _dailyLbls[i];
label.x = labelXPos;
label.y = 406;

if ((i + 1) != _claimDay || _isClaimed)
bitmapXPos += 81;
else
bitmapXPos += 70;

labelXPos += 82;
}

_dailyBtn.x = 43 + _claimDay * 81;
_dailyBtn.y = 398 - _dailyBtn.height;
}

private function update( e:TimerEvent ):void
{
_isClaimed = false;
_claimDay++;
if (_dailyRewardModel.resetTimeRemainingMS <= 0 || _claimDay > 6)
_claimDay = 0;

var chestBitmap:Bitmap;
var checkboxBitmap:Bitmap;
var dayLabel:Label;
var chestBitmapName:String;
var checkboxBitmapName:String;
for (var i:uint = 0; i < MAX_DAILY_REWARDS; ++i)
{
chestBitmap = _dailyIcons[i];
checkboxBitmap = _dailyCheckbox[i];
dayLabel = _dailyLbls[i];

if (i > _claimDay)
{
chestBitmapName = 'LoginCrate' + i + 'ClosedBMD';
checkboxBitmapName = 'BtnCheckBoxUpBMD';
chestBitmap.alpha = 1.0;
dayLabel.textColor = 0xd1e5f7;
dayLabel.setTextWithTokens(_dayText, {'[[Number.Day]]':i});
} else if (i < _claimDay)
{
chestBitmapName = 'LoginCrate' + i + 'EmptyBMD';
checkboxBitmapName = 'BtnCheckBoxSelectedBMD';
chestBitmap.alpha = 0.5;
dayLabel.textColor
```

```
= 0x213745;
dayLabel.setTextWithTokens(_dayText, {'[[Number.Day]]':i});
} else
{
if (!_isClaimed)
{
chestBitmapName = 'LoginCrate' + i + 'ClaimBMD'
checkboxBitmapName = 'BtnCheckBoxUpBMD';
chestBitmap.alpha = 1.0;
dayLabel.textColor = 0xffcf4f;
dayLabel.text = _todayText;
} else
{
chestBitmapName = 'LoginCrate' + i + 'EmptyBMD';
checkboxBitmapName = 'BtnCheckBoxSelectedBMD';
chestBitmap.alpha = 0.5;
dayLabel.textColor = 0x213745;
dayLabel.setTextWithTokens(_dayText, {'[[Number.Day]]':i});
}
}

chestBitmap.bitmapData = UIFactory.getBitmapData(chestBitmapName);
checkboxBitmap.bitmapData = UIFactory.getBitmapData(checkboxBitmapName);
}

_currentChestGlow.bitmapData = UIFactory.getBitmapData('LoginCrate' + _claimDay +
'GlowBMD');
_currentChestGlow.alpha = 0;

if (!_isClaimed)
{
_dailyBtn.enabled = true;
addListener(_dailyBtn, MouseEvent.CLICK, claimReward);
addListener(_dailyBtn, MouseEvent.ROLL_OVER, onChestRollOver);
addListener(_dailyBtn, MouseEvent.ROLL_OUT, onChestRollOut);
onFadeOut(_currentChestGlow);
}

layoutRewards();
}

private function onChestRollOver( e:MouseEvent = null ):void
{
if (_currentChestGlow)
{
TweenLite.killTweensOf(_currentChestGlow);
_currentChestGlow.alpha = 1;
}
}

private
```

```
function onChestRollOut( e:MouseEvent = null ):void
{
if (_currentChestGlow)
{
_currentChestGlow.alpha = 0;
onFadeOut(_currentChestGlow);
}
}

private function onFadeOut( fadeBitmap:Bitmap ):void
{
TweenLite.to(fadeBitmap, 0.65, {alpha:0.8, ease:Quad.easeOut, onComplete:onFadeIn,
onCompleteParams:[fadeBitmap], overwrite:0});
}

private function onFadeIn( fadeBitmap:Bitmap ):void
{
TweenLite.to(fadeBitmap, 0.65, {alpha:0.4, ease:Quad.easeIn, onComplete:onFadeOut,
onCompleteParams:[fadeBitmap], overwrite:0});
}

private function onPipClicked( p:PipEvent ):void
{
showMessageByIdx(p.index);
}

private function showMessageByIdx( idx:int ):void
{
_messageTitle.text = _messageVOs[idx].title;
_message.htmlText = _messageVOs[idx].message;

presenter.loadIcon(_messageVOs[idx].imageURL, _image.onImageLoaded);
_messageBG.y = 286;
_message.y = 210;
_messageBG.height = _message.textHeight + 20 < 73 ? _message.textHeight + 20 : 73;

_scrollRect.height = _messageBG.height - 15;
_scrollbar.updateScrollbarHeight(_scrollRect.height);
_scrollbar.updateScrollableHeight(_message.textHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.y = _messageBG.y + 10;
_scrollbar.resetScroll();

if (!_messageVOs[idx].isRead)
{
_messageVOs[idx].isRead = true;
presenter.sendMotDMessageRead(_messageVOs[idx].key);
}

if (_pipComponent.visible)
_pipComponent.setPipState(idx,
```

```
_messageVOs[idx].isRead);
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_message.textHeight - _scrollRect.height) * percent;
_message.scrollRect = _scrollRect;
}

private function claimReward( e:MouseEvent ):void
{
_isClaimed = true;
removeListener(_dailyBtn, MouseEvent.CLICK, claimReward);
removeListener(_dailyBtn, MouseEvent.ROLL_OVER, onChestRollOver);
removeListener(_dailyBtn, MouseEvent.ROLL_OUT, onChestRollOut);

presenter.sendDailyClaimRequest(_dailyRewardModel.header, _dailyRewardModel.protocolID);
}

private function showRewardView( rewards:StarbaseDailyRewardResponse ):void
{
var view:DailyRewardView = DailyRewardView(_viewFactory.createView(DailyRewardView));
view.rewards = rewards;
_viewFactory.notify(view);

destroy();
}

private function addBitmap( className:String ):Bitmap
{
var newBmp:Bitmap = UIFactory.getBitmap(className);
addChild(newBmp);
return newBmp;
}

private function addDayLabel( day:int ):Label
{
var dayLbl:Label = new Label(22, 0xd1e5f7, 60, 30, true);
dayLbl.constrictTextToSize = false;
dayLbl.align = TextFormatAlign.CENTER;
if (_claimDay > (day - 1))
{
dayLbl.textColor = 0x213745;
dayLbl.setTextWithTokens(_dayText, {'[[Number.Day]]':day});
} else if (_claimDay < (day - 1))
{
dayLbl.textColor = 0xd1e5f7;
dayLbl.setTextWithTokens(_dayText, {'[[Number.Day]]':day});
} else
{
dayLbl.textColor
```

```
= 0xffcf4f;
dayLbl.text = _todayText;
}
addChild(dayLbl);

return dayLbl;
}

override protected function onClose( e:MouseEvent = null ):void
{
if (presenter.currentGameState == StateEvent.GAME_STARBASE)
presenter.dispatch(new StarbaseEvent(StarbaseEvent.WELCOME_BACK));

super.onClose(e);
}

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function destroy():void
{
presenter.removeDailyRewardListener(showRewardView);
super.destroy();

_bg = null;
_messageBG = null;
_scrollRect = null;
_messageVOs = null;

if (_currentChestGlow)
TweenLite.killTweensOf(_currentChestGlow);

_currentChestGlow = null;

if (_image)
ObjectPool.give(_image);
_image = null;

if (_message)
_message.destroy();
_message = null;

if (_windowTitle)
_windowTitle.destroy();
_windowTitle = null;

if
```

```
(_messageTitle)
_messageTitle.destroy();
_messageTitle = null;

if (_dailyHeader)
_dailyHeader.destroy();
_dailyHeader = null;

if (_scrollbar)
_scrollbar.destroy();
_scrollbar = null;


if (_pipComponent)
_pipComponent.destroy();
_pipComponent = null;

if (_dailyBtn)
_dailyBtn.destroy();
_dailyBtn = null;


if (_dailyIcons)
{
var chestBitmap:Bitmap = _dailyIcons[_claimDay];
if (chestBitmap)
TweenLite.killTweensOf(chestBitmap);
_dailyIcons.length = 0;
}
_dailyIcons = null;

if (_dailyCheckbox)
_dailyCheckbox.length = 0;

_dailyCheckbox = null;

if (_dailyLbls)
_dailyLbls.length = 0;

_dailyLbls = null;

if (_dailySideMessage)
_dailySideMessage.destroy();

_dailySideMessage = null;

_loginBonusProto = null;

if (_claimTimer)
_claimTimer.stop();

_claimTimer
```

```
= null;
}
}
}

-----------------------------------
File 821: igw\com\ui\modal\information\ResourceModalView.as
package com.ui.modal.information
{
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.adobe.utils.StringUtil;

public class ResourceModalView extends View
{
private var _bg:Sprite;
private var _actionBtn:BitmapButton;
private var _cancelBtn:BitmapButton;
private var _closeBtn:BitmapButton;

private var _alloySymbol:Bitmap;
private var _creditsSymbol:Bitmap;
private var _energySymbol:Bitmap;
private var _syntheticsSymbol:Bitmap;
private var _premiumSymbol:Bitmap;

private var _actionCostLbl:Label;
private var _windowTitle:Label;
private var _windowSubTitle:Label;
private var _alloyLbl:Label;
private var _creditsLbl:Label;
private var _energyLbl:Label;
private var _syntheticsLbl:Label;

private var _actionBtnText:String;

private var _actionCallback:Function;

[PostConstruct]
```

```
override public function init():void
{
super.init();
}

public function setUp( creditAmount:int, alloyAmount:int, energyAmount:int, syntheticAmount:int,
windowTitle:String, windowSubtitle:String, isGainingResources:Boolean,
actionCallback:Function,
actionCost:Number = 0, actionBtnTxt:String = 'CodeString.BuildRecycle.Title.Recycle' ):void
{
_actionCallback = actionCallback;
_actionBtnText = actionBtnTxt;

var mcBGClass:Class = Class(getDefinitionByName('ResourceModalWindowMC'));
_bg = Sprite(new mcBGClass());
addChild(_bg);

_windowTitle = new Label(22, 0xFFFFFF, 383, 29, true);
_windowTitle.x = 18;
_windowTitle.y = 11.5;
_windowTitle.constrictTextToSize = false;
_windowTitle.align = TextFormatAlign.LEFT;
_windowTitle.text = windowTitle;
_windowTitle.letterSpacing = 1.5;
addChild(_windowTitle);

_windowSubTitle = new Label(22, 0xfbefaf, 360, 25, true);
_windowSubTitle.x = 27;
_windowSubTitle.y = 50;
_windowSubTitle.constrictTextToSize = false;
_windowSubTitle.align = TextFormatAlign.CENTER;
_windowSubTitle.text = windowSubtitle;
_windowSubTitle.letterSpacing = 1.5;
addChild(_windowSubTitle);

_creditsSymbol = PanelFactory.getPanel('LootedResourceCreditsBMD');
_creditsSymbol.x = 33;
_creditsSymbol.y = 87;
addChild(_creditsSymbol);

_alloySymbol = PanelFactory.getPanel('LootedResourceAlloyBMD');
_alloySymbol.x = _creditsSymbol.x + 179;
_alloySymbol.y = _creditsSymbol.y;
addChild(_alloySymbol);

_energySymbol = PanelFactory.getPanel('LootedResourceEnergyBMD');
_energySymbol.x = _creditsSymbol.x;
_energySymbol.y = _creditsSymbol.y + 50;
addChild(_energySymbol);

_syntheticsSymbol
```

```
= PanelFactory.getPanel('LootedResourceSyntheticsBMD');
_syntheticsSymbol.x = _alloySymbol.x;
_syntheticsSymbol.y = _energySymbol.y;
addChild(_syntheticsSymbol);

var textColor:uint;
var resourceLbl:String;
if (!isGainingResources && creditAmount > 0)
{
textColor = 0xf04c4c;
resourceLbl = '-' + StringUtil.commaFormatNumber(creditAmount);
} else
{
textColor = 0x7afe60;
resourceLbl = StringUtil.commaFormatNumber(creditAmount);
}
_creditsLbl = new Label(16, textColor, 100, 30, true, 1);
_creditsLbl.x = _creditsSymbol.width * 0.5 + _creditsSymbol.x - 37;
_creditsLbl.y = _creditsSymbol.height * 0.5 + _creditsSymbol.y - 6;
_creditsLbl.constrictTextToSize = false;
_creditsLbl.align = TextFormatAlign.LEFT;
_creditsLbl.autoSize = TextFieldAutoSize.LEFT;
_creditsLbl.text = resourceLbl;
_creditsLbl.letterSpacing = 1.5;
addChild(_creditsLbl);

if (!isGainingResources && alloyAmount > 0)
{
textColor = 0xf04c4c;
resourceLbl = '-' + StringUtil.commaFormatNumber(alloyAmount);
} else
{
textColor = 0x7afe60;
resourceLbl = StringUtil.commaFormatNumber(alloyAmount);
}
_alloyLbl = new Label(16, textColor, 100, 30, true, 1);
_alloyLbl.x = _alloySymbol.width * 0.5 + _alloySymbol.x - 37;
_alloyLbl.y = _alloySymbol.height * 0.5 + _alloySymbol.y - 6;
_alloyLbl.constrictTextToSize = false;
_alloyLbl.align = TextFormatAlign.LEFT;
_alloyLbl.autoSize = TextFieldAutoSize.LEFT;
_alloyLbl.text = resourceLbl;
_alloyLbl.letterSpacing = 1.5;
addChild(_alloyLbl);

if (!isGainingResources && syntheticAmount > 0)
{
textColor = 0xf04c4c;
resourceLbl = '-' + StringUtil.commaFormatNumber(syntheticAmount);
} else
{
```

```
textColor = 0x7afe60;
resourceLbl = StringUtil.commaFormatNumber(syntheticAmount);
}
_syntheticsLbl = new Label(16, textColor, 100, 30, true, 1);
_syntheticsLbl.x = _syntheticsSymbol.width * 0.5 + _syntheticsSymbol.x - 37;
_syntheticsLbl.y = _syntheticsSymbol.height * 0.5 + _syntheticsSymbol.y - 6;
_syntheticsLbl.constrictTextToSize = false;
_syntheticsLbl.align = TextFormatAlign.LEFT;
_syntheticsLbl.autoSize = TextFieldAutoSize.LEFT;
_syntheticsLbl.text = resourceLbl;
_syntheticsLbl.letterSpacing = 1.5;
addChild(_syntheticsLbl);

if (!isGainingResources && energyAmount > 0)
{
textColor = 0xf04c4c;
resourceLbl = '-' + StringUtil.commaFormatNumber(energyAmount);
} else
{
textColor = 0x7afe60;
resourceLbl = StringUtil.commaFormatNumber(energyAmount);
}
_energyLbl = new Label(16, textColor, 100, 30, true, 1);
_energyLbl.x = _energySymbol.width * 0.5 + _energySymbol.x - 37;
_energyLbl.y = _energySymbol.height * 0.5 + _energySymbol.y - 6;
_energyLbl.constrictTextToSize = false;
_energyLbl.align = TextFormatAlign.LEFT;
_energyLbl.autoSize = TextFieldAutoSize.LEFT;
_energyLbl.text = resourceLbl;
_energyLbl.letterSpacing = 1.5;
addChild(_energyLbl);

_cancelBtn = ButtonFactory.getBitmapButton('RedBtnNeutralBMD', 61, 200,
'CodeString.Shared.CancelBtn', 0xF58993, 'RedBtnRolloverBMD', 'RedBtnSelectedBMD');
_cancelBtn.fontSize = 26;
_cancelBtn.label.y += 5;
_cancelBtn.addEventListener(MouseEvent.CLICK, onCancelBtnClicked);
addChild(_cancelBtn);

_closeBtn = ButtonFactory.getCloseButton(_bg.width - 36, 11);
addListener(_closeBtn, MouseEvent.CLICK, onClose);
addChild(_closeBtn);

addActionButton(isGainingResources, actionCost);

addEffects();
effectsIN();
}

private
```

```
function addActionButton( isGainingResources:Boolean, actionCost:Number = 0 ):void
{
if (isGainingResources)
{
_actionBtn = ButtonFactory.getBitmapButton('BlueBtnNeutralBMD', 206, 200, _actionBtnText,
0xc9e6f6, 'BlueBtnRolloverBMD', 'BlueBtnSelectedBMD');
_actionBtn.fontSize = 26;
_actionBtn.label.y += 5;
//_actionBtn.label.constrictTextToSize = false;
} else
{
var premiumSymbolClass:Class = Class(getDefinitionByName(('KalganSymbolBMD')));
_premiumSymbol = new Bitmap(BitmapData(new premiumSymbolClass()));
_premiumSymbol.x = 240;
_premiumSymbol.y = 229;

_actionCostLbl = new Label(18, 0xf0f0f0, 50, 25, false);
_actionCostLbl.align = TextFormatAlign.LEFT;
_actionCostLbl.autoSize = TextFieldAutoSize.LEFT;
_actionCostLbl.constrictTextToSize = false;
_actionCostLbl.x = 279;
_actionCostLbl.y = 229;

if (actionCost > 0)
{
_actionCostLbl.useLocalization = false;
_actionCostLbl.text = String(actionCost);
} else
{
_actionCostLbl.useLocalization = true;
_actionCostLbl.text = 'CodeString.Shared.Free';
}

_actionBtn = ButtonFactory.getBitmapButton('btnBuyNeutralBMD', 206, 200,
'CodeString.Shared.GetResources', 0xf7c78b, 'btnBuyRolloverBMD', 'btnBuyNeutralBMD');
_actionBtn.fontSize = 20;
_actionBtn.label.constrictTextToSize = false;
_actionBtn.label.x += 5;
_actionBtn.label.y -= 4;
}

addListener(_actionBtn, MouseEvent.CLICK, onActionBtnClicked);
addChild(_actionBtn);

if (_actionCostLbl)
{
addChild(_actionCostLbl);
addChild(_premiumSymbol);
}
}

private
```

```
function onActionBtnClicked( e:MouseEvent ):void
{
//Use the callback here so this window is destroyed properly
if (_actionCallback != null)
_actionCallback();

destroy();
}

private function onCancelBtnClicked( e:MouseEvent ):void
{
destroy();
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function destroy():void
{
super.destroy();
_bg = null;
_closeBtn.destroy();
_closeBtn = null;

_actionBtn.destroy();
_actionBtn = null;
_cancelBtn.destroy();
_cancelBtn = null;

_alloySymbol = null;
_creditsSymbol = null;
_energySymbol = null;
_syntheticsSymbol = null;

if (_actionCostLbl)
{
_actionCostLbl.destroy();
_actionCostLbl = null;
}

_windowTitle.destroy();
_windowTitle = null;

_windowSubTitle.destroy();
_windowSubTitle = null;

_alloyLbl.destroy();
_alloyLbl = null;

_creditsLbl.destroy();
_creditsLbl
```

```
= null;

_energyLbl.destroy();
_energyLbl = null;

_syntheticsLbl.destroy();
_syntheticsLbl = null;
}
}
}

-----------------------------------
File 822: igw\com\ui\modal\information\StatInformationView.as
package com.ui.modal.information
{
import com.enum.server.PurchaseTypeEnum;
import com.event.TransactionEvent;
import com.model.starbase.BuildingVO;
import com.presenter.starbase.IStarbasePresenter;
import com.ui.core.ButtonPrototype;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.label.LabelFactory;
import com.ui.core.component.misc.TooltipComponent;
import com.ui.modal.ButtonFactory;

import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

public class StatInformationView extends View
{
private const DIALOG_MARGIN:int = 120;

private var _bg:Sprite;
private var _title:Label;
private var _body:Label;

private var _btnX:Number;
private var _btnY:Number;
private var _building:BuildingVO;
private var _bmOkBtn:BitmapButton;
private var _bmCancelBtn:BitmapButton;

private var _buttonProtos:Vector.<ButtonPrototype>;

[PostConstruct]
public override function init():void
{
```

```
super.init();
_buttonProtos = new Vector.<ButtonPrototype>;

buildBtns(_btnX, _btnY);

addEffects();
effectsIN();
}

public function SetUp( toolTip:TooltipComponent, title:String = 'CodeString.Shared.Stats',
btnX:Number = 235, btnY:Number = 50, bg:String = 'StatWindowBGMC' ):void
{
var windowBG:Class = Class(getDefinitionByName(bg));
_bg = Sprite(new windowBG());
addChild(_bg);

_title = LabelFactory.createLabel(LabelFactory.LABEL_TYPE_DIALOG_TITLE, _bg.width -
DIALOG_MARGIN);
_title.align = TextFormatAlign.LEFT;
_title.x = 20;
_title.y = 12;
_title.text = title;
addChild(_title);

addChild(toolTip);

_btnX = btnX;
_btnY = btnY;
}

private function onRecycleClick( e:MouseEvent ):void
{
//trace('recycled');
presenter.performTransaction(TransactionEvent.STARBASE_BUILDING_RECYCLE, _building,
PurchaseTypeEnum.INSTANT);
onButtonClicked(null);
}

private function buildBtns( btnX:Number, btnY:Number ):void
{
// _okBtn = new ButtonPrototype('CodeString.Shared.OkBtn');
// addButton(_okBtn.text, SIZE_SMALL, _bg.width - btnX, _bg.height - 10);
_bmOkBtn = ButtonFactory.getBitmapButton('MiddleBtnUpBMD', 127, 356,
'CodeString.Shared.OkBtn', 0xc9e6f6, 'MiddleBtnRollOverBMD', '', '', 'MiddleBtnDownBMD', 0,
0);
//_bmOkBtn.fontSize = 26;
addChild(_bmOkBtn);
addListener(_bmOkBtn, MouseEvent.CLICK, onButtonClicked);
}

protected
```

```
function onButtonClicked( e:MouseEvent ):int
{
// var idx:int = super.onButtonClicked(e);
//
// var proto:ButtonPrototype = _buttonProtos[idx];
//
// if (proto.callback != null)
// proto.callback.apply(null, proto.args);
//
// if (proto.doClose)
destroy();

return 0;
}

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

public function get building():BuildingVO { return _building; }
public function set building( value:BuildingVO ):void { _building = value; }

[Inject]
public function set presenter( value:IStarbasePresenter ):void { _presenter = value; }
public function get presenter():IStarbasePresenter { return IStarbasePresenter(_presenter); }

override public function destroy():void
{
super.destroy();
_bg = null;
_title.destroy();
_title = null;
_bmOkBtn.destroy();
_bmOkBtn = null;
if (_bmCancelBtn)
{
_bmCancelBtn.destroy();
_bmCancelBtn = null;
}
}
}
}

---
File 823: igw\com\ui\modal\intro\FactionSelectView.as
ï»¿package com.ui.modal.intro
{
import com.Application;
import com.controller.sound.SoundController;
import com.enum.AudioEnum;
import com.enum.FactionEnum;
import
```

```
com.enum.TimeLogEnum;
import com.event.LoadEvent;
import com.model.player.CurrentUser;
import com.presenter.preload.IPreloadPresenter;
import com.service.language.Localization;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.core.component.videoplayer.YouTubeVideoPlayer;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.ui.modal.intro.characterselect.CharacterSelectView;
import com.util.CommonFunctionUtil;
import com.util.TimeLog;

import flash.display.Bitmap;
import flash.display.Sprite;
import flash.events.Event;
import flash.events.IEventDispatcher;
import flash.events.MouseEvent;
import flash.filters.GlowFilter;
import flash.text.StyleSheet;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;

import flash.external.*;
import flash.net.*;
import flash.display.*;
import flash.net.*;

import org.parade.util.DeviceMetrics;

public class FactionSelectView extends View
{
public static var ANALYTICS_FIRST_TIME:Boolean = true;

private var _bg:Bitmap;
private var _eventDispatcher:IEventDispatcher;
private var _title:Label;
private var _choose:Label;

private var _sovBG:Sprite;
private var _tyrBG:Sprite;
private var _igaBG:Sprite;

private var _sovTitle:Label;
private var _tyrTitle:Label;
private var _igaTitle:Label;

private var _sovFlavor:Label;
private
```

```
var _tyrFlavor:Label;
private var _igaFlavor:Label;

private var _sovDescription:Label;
private var _tyrDescription:Label;
private var _igaDescription:Label;

private var _sovVideo:BitmapButton;
private var _tyrVideo:BitmapButton;
private var _igaVideo:BitmapButton;

private var _sovSelect:BitmapButton;
private var _tyrSelect:BitmapButton;
private var _igaSelect:BitmapButton;

private var _glowFilter:GlowFilter;
private var _youTubeVideo:YouTubeVideoPlayer;
private var _closeVideoBtn:BitmapButton;
private var _videoBG:Sprite;

private var _factionNameSov:String = 'CodeString.FactionSelect.FactionName.Sov';
private var _factionNameTyr:String = 'CodeString.FactionSelect.FactionName.Tyrannar';
private var _factionNameIGA:String = 'CodeString.FactionSelect.FactionName.IGA';

private var _factionDescriptionSov:String = 'CodeString.FactionSelect.FactionDescription.Sov';
private var _factionDescriptionTyr:String =
'CodeString.FactionSelect.FactionDescription.Tyrannar';
private var _factionDescriptionIGA:String = 'CodeString.FactionSelect.FactionDescription.IGA';

private var _factionFlavorTextSov:String = 'CodeString.FactionSelect.FlavorText.Sov';
private var _factionFlavorTextTyr:String = 'CodeString.FactionSelect.FlavorText.Tyrannar';
private var _factionFlavorTextIGA:String = 'CodeString.FactionSelect.FlavorText.IGA';

private var _titleText:String = 'CodeString.FactionSelect.Title';
private var _subTitleText:String = 'CodeString.FactionSelect.SubTitle';

private var _selectBtnText:String = 'CodeString.FactionSelect.SelectBtn';
private var _videoBtnText:String = 'CodeString.FactionSelect.VideoBtn';

[PostConstruct]
override public function init():void
{
super.init();

TimeLog.startTimeLog(TimeLogEnum.FACTION_SELECT);
if (ANALYTICS_FIRST_TIME)
{
ANALYTICS_FIRST_TIME = false;
presenter.trackPlayerProgress(100000);
}

_bg
```

```
= PanelFactory.getPanel('PreloadBGBMD');
_glowFilter = CommonFunctionUtil.createGlow(0x73bdf0, true, 25, 25);

_title = new Label(30, 0xf0f0f0);
_title.constrictTextToSize = false;
_title.allCaps = true;
_title.autoSize = TextFieldAutoSize.CENTER;
_title.text = "Loading...";
_title.x = (_bg.width - _title.width) * .5;
_title.y = (_bg.height - _title.height) * .5;

addChild(_bg);
addChild(_title);

if (Localization.loaded)
layout();
else
_eventDispatcher.addEventListener(LoadEvent.LOCALIZATION_COMPLETE, layout);
addListener(Application.STAGE, Event.RESIZE, onResize);

addEffects();
effectsIN();
}

private function layout( e:Event = null ):void
{
var style:StyleSheet = new StyleSheet();
var capsFirst:Object = new Object();
capsFirst.fontSize = "24";
capsFirst.display = 'inline';
style.setStyle("CapsFirst", capsFirst);

_title.autoSize = TextFieldAutoSize.LEFT;
_title.x = 37;
_title.y = 9;
_title.text = _titleText;

_choose = new Label(26, 0xfbefaf);
_choose.constrictTextToSize = false;
_choose.allCaps = true;
_choose.autoSize = TextFieldAutoSize.LEFT;
_choose.x = _title.x + _title.textWidth + 9;
_choose.y = 13;
_choose.text = _subTitleText;

var sovBitmap:Bitmap = PanelFactory.getPanel('FactionSOVBMD');
_sovBG = new Sprite();
_sovBG.addChild(sovBitmap);
_sovBG.buttonMode = true;
_sovBG.useHandCursor = true;
_sovBG.x
```

```
= 28;
_sovBG.y = 55;
addListener(_sovBG, MouseEvent.CLICK, onButtonClick);
addListener(_sovBG, MouseEvent.ROLL_OVER, onButtonRollover);
addListener(_sovBG, MouseEvent.ROLL_OUT, onButtonRollout);

var tyrBitmap:Bitmap = PanelFactory.getPanel('FactionTYRBMD');
_tyrBG = new Sprite();
_tyrBG.addChild(tyrBitmap);
_tyrBG.buttonMode = true;
_tyrBG.useHandCursor = true;
_tyrBG.x = 338;
_tyrBG.y = 55;
addListener(_tyrBG, MouseEvent.CLICK, onButtonClick);
addListener(_tyrBG, MouseEvent.ROLL_OVER, onButtonRollover);
addListener(_tyrBG, MouseEvent.ROLL_OUT, onButtonRollout);

var igaBitmap:Bitmap = PanelFactory.getPanel('FactionIGABMD');
_igaBG = new Sprite();
_igaBG.addChild(igaBitmap);
_igaBG.buttonMode = true;
_igaBG.useHandCursor = true;
_igaBG.x = 648;
_igaBG.y = 55;
addListener(_igaBG, MouseEvent.CLICK, onButtonClick);
addListener(_igaBG, MouseEvent.ROLL_OVER, onButtonRollover);
addListener(_igaBG, MouseEvent.ROLL_OUT, onButtonRollout);

_sovTitle = new Label(18, 0xd1e5f7, 296, 32, true, 1);
_sovTitle.constrictTextToSize = false;
_sovTitle.styleSheet = style;
_sovTitle.x = _sovBG.x;
_sovTitle.y = _sovBG.y - 5;
_sovTitle.htmlText = _factionNameSov;

_tyrTitle = new Label(18, 0xd1e5f7, 296, 32, true, 1);
_tyrTitle.constrictTextToSize = false;
_tyrTitle.styleSheet = style;
_tyrTitle.x = _tyrBG.x;
_tyrTitle.y = _tyrBG.y - 5;
_tyrTitle.htmlText = _factionNameTyr;

_igaTitle = new Label(18, 0xd1e5f7, 296, 32, true, 1);
_igaTitle.constrictTextToSize = false;
_igaTitle.styleSheet = style;
_igaTitle.x = _igaBG.x;
_igaTitle.y = _igaBG.y - 5;
_igaTitle.htmlText = _factionNameIGA;

_sovFlavor = new Label(28, 0xf0f0f0, 207, 30);
_sovFlavor.constrictTextToSize
```

```
= false;
_sovFlavor.align = TextFormatAlign.LEFT;
_sovFlavor.x = 121;
_sovFlavor.y = 458;
_sovFlavor.text = _factionFlavorTextSov;

_tyrFlavor = new Label(28, 0xf0f0f0, 207, 30);
_tyrFlavor.constrictTextToSize = false;
_tyrFlavor.align = TextFormatAlign.LEFT;
_tyrFlavor.x = 426;
_tyrFlavor.y = 458;
_tyrFlavor.text = _factionFlavorTextTyr;

_igaFlavor = new Label(28, 0xf0f0f0, 207, 30);
_igaFlavor.constrictTextToSize = false;
_igaFlavor.align = TextFormatAlign.LEFT;
_igaFlavor.x = 738;
_igaFlavor.y = 458;
_igaFlavor.text = _factionFlavorTextIGA;

_sovDescription = new Label(12, 0xf0f0f0, 268, 61, true, 1);
_sovDescription.constrictTextToSize = false;
_sovDescription.multiline = true;
_sovDescription.align = TextFormatAlign.LEFT;
_sovDescription.x = 38;
_sovDescription.y = 495;
_sovDescription.text = _factionDescriptionSov;

_tyrDescription = new Label(12, 0xf0f0f0, 268, 61, true, 1);
_tyrDescription.constrictTextToSize = false;
_tyrDescription.multiline = true;
_tyrDescription.align = TextFormatAlign.LEFT;
_tyrDescription.x = 350;
_tyrDescription.y = 495;
_tyrDescription.text = _factionDescriptionTyr;

_igaDescription = new Label(12, 0xf0f0f0, 268, 61, true, 1);
_igaDescription.constrictTextToSize = false;
_igaDescription.multiline = true;
_igaDescription.align = TextFormatAlign.LEFT;
_igaDescription.x = 656;
_igaDescription.y = 495;
_igaDescription.text = _factionDescriptionIGA;

_sovVideo = ButtonFactory.getBitmapButton('PreloadGenericBtnUpBMD', 31, 565,
_videoBtnText, 0xf0f0f0, 'PreloadGenericBtnRollOverBMD', 'PreloadGenericBtnDownBMD', null,
'PreloadGenericBtnDownBMD');
addListener(_sovVideo, MouseEvent.CLICK, onButtonClick);

_sovSelect = ButtonFactory.getBitmapButton('PreloadGenericBtnUpBMD', 185, 565,
_selectBtnText,
```

```
0xf0f0f0, 'PreloadGenericBtnRollOverBMD', 'PreloadGenericBtnDownBMD', null,
'PreloadGenericBtnDownBMD');
addListener(_sovSelect, MouseEvent.CLICK, onButtonClick);
addListener(_sovSelect, MouseEvent.ROLL_OVER, onButtonRollover);
addListener(_sovSelect, MouseEvent.ROLL_OUT, onButtonRollout);

_tyrVideo = ButtonFactory.getBitmapButton('PreloadGenericBtnUpBMD', 339, 565,
_videoBtnText, 0xf0f0f0, 'PreloadGenericBtnRollOverBMD', 'PreloadGenericBtnDownBMD', null,
'PreloadGenericBtnDownBMD');
addListener(_tyrVideo, MouseEvent.CLICK, onButtonClick);

_tyrSelect = ButtonFactory.getBitmapButton('PreloadGenericBtnUpBMD', 493, 565,
_selectBtnText, 0xf0f0f0, 'PreloadGenericBtnRollOverBMD', 'PreloadGenericBtnDownBMD',
null, 'PreloadGenericBtnDownBMD');
addListener(_tyrSelect, MouseEvent.CLICK, onButtonClick);
addListener(_tyrSelect, MouseEvent.ROLL_OVER, onButtonRollover);
addListener(_tyrSelect, MouseEvent.ROLL_OUT, onButtonRollout);

_igaVideo = ButtonFactory.getBitmapButton('PreloadGenericBtnUpBMD', 650, 565,
_videoBtnText, 0xf0f0f0, 'PreloadGenericBtnRollOverBMD', 'PreloadGenericBtnDownBMD', null,
'PreloadGenericBtnDownBMD');
addListener(_igaVideo, MouseEvent.CLICK, onButtonClick);

_igaSelect = ButtonFactory.getBitmapButton('PreloadGenericBtnUpBMD', 804, 565,
_selectBtnText, 0xf0f0f0, 'PreloadGenericBtnRollOverBMD', 'PreloadGenericBtnDownBMD',
null, 'PreloadGenericBtnDownBMD');
addListener(_igaSelect, MouseEvent.CLICK, onButtonClick);
addListener(_igaSelect, MouseEvent.ROLL_OVER, onButtonRollover);
addListener(_igaSelect, MouseEvent.ROLL_OUT, onButtonRollout);

_youTubeVideo = new YouTubeVideoPlayer(915, 549, true);
_youTubeVideo.x = (DeviceMetrics.WIDTH_PIXELS - _bg.width) * 0.5 + 29;
_youTubeVideo.y = (DeviceMetrics.HEIGHT_PIXELS - _bg.height) * 0.5 + 55;
_youTubeVideo.onFullScreenChanged.add(onVideoFullScreen);
_youTubeVideo.onVideoEnd = onVideoEnd;
_youTubeVideo.visible = false;

_closeVideoBtn = ButtonFactory.getCloseButton(_bg.x + _bg.width - 40, _bg.y + 16);
_closeVideoBtn.visible = false;
addListener(_closeVideoBtn, MouseEvent.CLICK, onCloseVideo);

_videoBG = new Sprite();
_videoBG.graphics.beginFill(0x000000, 1.0);
_videoBG.graphics.drawRect(30, 57, 915, 549);
_videoBG.graphics.endFill();
_videoBG.visible = false;

addChild(_title);
addChild(_choose);
addChild(_sovBG);
addChild(_tyrBG);
```

```
addChild(_igaBG);
addChild(_sovTitle);
addChild(_tyrTitle);
addChild(_igaTitle);
addChild(_sovFlavor);
addChild(_tyrFlavor);
addChild(_igaFlavor);
addChild(_sovDescription);
addChild(_tyrDescription);
addChild(_igaDescription);
addChild(_sovVideo);
addChild(_tyrVideo);
addChild(_igaVideo);
addChild(_sovSelect);
addChild(_tyrSelect);
addChild(_igaSelect);

Application.STAGE.addChild(_youTubeVideo);
addChild(_closeVideoBtn);
addChild(_videoBG);
}

private function onButtonClick( e:MouseEvent ):void
{
switch (e.target)
{
case _sovVideo:
if(CONFIG::IS_DESKTOP)
{
var urlRequest:URLRequest = new
URLRequest("https://www.youtube.com/embed/vpwvNWG06jc");
navigateToURL(urlRequest);
}
else
ExternalInterface.call("vidinjector(\"https://www.youtube.com/embed/vpwvNWG06jc\",80)");
//todo uncomment later
//openVideo('e6nEVTVtUzs', 57);
break;
case _tyrVideo:
if(CONFIG::IS_DESKTOP)
{
var urlRequest:URLRequest = new
URLRequest("https://www.youtube.com/embed/-006qEfda-c");
navigateToURL(urlRequest);
}
else
ExternalInterface.call("vidinjector(\"https://www.youtube.com/embed/-006qEfda-c\",80)");
//navigateToURL("https://www.youtube.com/embed/-006qEfda-c");
//todo uncomment later
//openVideo('Gdz4VDXCl5M',
```

```
41);
break;
case _igaVideo:
if(CONFIG::IS_DESKTOP)
{
var urlRequest:URLRequest = new
URLRequest("https://www.youtube.com/embed/4r__6056VJg");
navigateToURL(urlRequest);
}
else
ExternalInterface.call("vidinjector(\"https://www.youtube.com/embed/4r__6056VJg\",80)");
//navigateToURL("https://www.youtube.com/embed/4r__6056VJg");
//todo uncomment later
//openVideo('SImP_rSg3KE', 61);
break;
case _sovBG:
case _sovSelect:
selectFaction(FactionEnum.SOVEREIGNTY);
break;
case _tyrBG:
case _tyrSelect:
selectFaction(FactionEnum.TYRANNAR);
break;
case _igaBG:
case _igaSelect:
selectFaction(FactionEnum.IGA);
break;
}
}

private function onButtonRollover( e:MouseEvent ):void
{
switch (e.target)
{
case _sovBG:
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_2, 0.5);
case _sovSelect:
_sovBG.filters = [_glowFilter];
_sovSelect.state = "over";
break;
case _igaBG:
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_2, 0.5);
case _igaSelect:
_igaBG.filters = [_glowFilter];
_igaSelect.state = "over";
break;
case _tyrBG:
SoundController.instance.playSound(AudioEnum.AFX_MOUSE_DOWN_CLICK_2, 0.5);
case _tyrSelect:
_tyrBG.filters = [_glowFilter];
_tyrSelect.state
```

```
= "over";
break;
}
}

private function onButtonRollout( e:MouseEvent ):void
{
switch (e.target)
{
case _sovBG:
case _sovSelect:
_sovBG.filters = [];
_sovSelect.state = "normal";
break;
case _igaBG:
case _igaSelect:
_igaBG.filters = [];
_igaSelect.state = "normal";
break;
case _tyrBG:
case _tyrSelect:
_tyrBG.filters = [];
_tyrSelect.state = "normal";
break;
}
}

private function onVideoFullScreen( isFullScreen:Boolean ):void
{
if (isFullScreen)
{
if (_youTubeVideo)
{
_youTubeVideo.x = 0;
_youTubeVideo.y = 0;
}

if (_closeVideoBtn)
_closeVideoBtn.visible = false;

} else
{

if (_youTubeVideo)
{
_youTubeVideo.x = (DeviceMetrics.WIDTH_PIXELS - _bg.width) * 0.5 + 29;
_youTubeVideo.y = (DeviceMetrics.HEIGHT_PIXELS - _bg.height) * 0.5 + 55;
}

if (_closeVideoBtn)
_closeVideoBtn.visible
```

```
= true;
}
}

private function selectFaction( faction:String ):void
{
CurrentUser.faction = faction;
showView(CharacterSelectView);
destroy();
}

private function openVideo( video:String, volume:int ):void
{
if (_youTubeVideo)
{
_youTubeVideo.updateVideo(video);
_youTubeVideo.volume = volume;
_youTubeVideo.visible = true;
}

if (_videoBG)
_videoBG.visible = true;

if (_closeVideoBtn)
_closeVideoBtn.visible = true;
}

private function onVideoEnd():void
{
if (_videoBG)
_videoBG.visible = false;

if (_closeVideoBtn)
_closeVideoBtn.visible = false;

if (_youTubeVideo)
_youTubeVideo.visible = false;
}

private function onCloseVideo( e:MouseEvent ):void
{
if (_youTubeVideo)
_youTubeVideo.stopVideo();

onVideoEnd();
}

private function onResize( e:Event ):void
{
if (_youTubeVideo)
{
```

```
_youTubeVideo.x = (DeviceMetrics.WIDTH_PIXELS - _bg.width) * 0.5 + 29;
_youTubeVideo.y = (DeviceMetrics.HEIGHT_PIXELS - _bg.height) * 0.5 + 55;
}
}

[Inject]
public function set eventDispatcher( value:IEventDispatcher ):void { _eventDispatcher = value; }
[Inject]
public function set presenter( value:IPreloadPresenter ):void { _presenter = value; }
public function get presenter():IPreloadPresenter { return IPreloadPresenter(_presenter); }

override public function get height():Number { return _bg.height; }
override public function get width():Number { return _bg.width; }

override public function get typeUnique():Boolean { return false; }

override public function destroy():void
{
_eventDispatcher.removeEventListener(LoadEvent.LOCALIZATION_COMPLETE, layout);
TimeLog.endTimeLog(TimeLogEnum.FACTION_SELECT);
removeListener(Application.STAGE, Event.RESIZE, onResize);
_bg = null;
_eventDispatcher = null;

_title.destroy();
_title = null;

_choose.destroy();
_choose = null;

_sovBG = null;

_tyrBG = null;

_igaBG = null;

_sovTitle.destroy();
_sovTitle = null;

_tyrTitle.destroy();
_tyrTitle = null;

_igaTitle.destroy();
_igaTitle = null;

_sovFlavor.destroy();
_sovFlavor = null;

_tyrFlavor.destroy();
_tyrFlavor
```

```
= null;

_igaFlavor.destroy();
_igaFlavor = null;

_sovDescription.destroy();
_sovDescription = null;

_tyrDescription.destroy();
_tyrDescription = null;

_igaDescription.destroy();
_igaDescription = null;

_sovVideo.destroy();
_sovVideo = null;

_tyrVideo.destroy();
_tyrVideo = null;

_igaVideo.destroy();
_igaVideo = null;

_sovSelect.destroy();
_sovSelect = null;

_tyrSelect.destroy();
_tyrSelect = null;

_igaSelect.destroy();
_igaSelect = null;

Application.STAGE.removeChild(_youTubeVideo);
_youTubeVideo.destroy();
_youTubeVideo = null;

_glowFilter = null;
_videoBG = null;

_closeVideoBtn.destroy();
_closeVideoBtn = null;

super.destroy()
}
}
}

-----------------------------------
File 824: igw\com\ui\modal\intro\FAQSelectionComponent.as
package com.ui.modal.intro
{
```

```
import com.enum.ui.ButtonEnum;
import com.enum.ui.PanelEnum;
import com.ui.UIFactory;
import com.ui.core.ScaleBitmap;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;

import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFieldAutoSize;
import flash.text.TextFormatAlign;
import flash.utils.getDefinitionByName;

import org.greensock.TweenLite;
import org.greensock.easing.Quad;
import org.osflash.signals.Signal;

public class FAQSelectionComponent extends Sprite
{
private var _bg:ScaleBitmap;
private var _hitArea:Sprite;
private var _scrollRect:Rectangle;
private var _selection:*;
private var _extended:Boolean;
private var _filter:String;
private var _sort:Number;
private var _contractBtn:BitmapButton;
private var _expandBtn:BitmapButton;
private var _selectionName:Label;
private var _selectionInfo:Label;

public var onSizeUpdated:Signal;
public var onResizeFinish:Signal;
public var onClicked:Signal;

public function FAQSelectionComponent()
{
onClicked = new Signal(FAQSelectionComponent);
onSizeUpdated = new Signal();
onResizeFinish = new Signal(FAQSelectionComponent);

_bg = UIFactory.getScaleBitmap(PanelEnum.FAQ_SUBJECT_BG);

_scrollRect = new Rectangle(0, 0, _bg.width, _bg.height);
this.scrollRect = _scrollRect;

_selectionName = new Label(20, 0xfbefaf, 90, 20, true);
_selectionName.allCaps
```

```
= true;
_selectionName.autoSize = TextFieldAutoSize.LEFT;
_selectionName.constrictTextToSize = false;

_selectionInfo = new Label(13, 0xf0f0f0, 485, 30, true, 1);
_selectionInfo.constrictTextToSize = false;
_selectionInfo.multiline = true;
_selectionInfo.autoSize = TextFieldAutoSize.LEFT;
_selectionInfo.align = TextFormatAlign.LEFT;

_contractBtn = UIFactory.getButton(ButtonEnum.FAQ_DOWN_ARROW, 0, 0, 503);
_contractBtn.addEventListener(MouseEvent.CLICK, onSelectionClick, false, 0, true);

_expandBtn = UIFactory.getButton(ButtonEnum.FAQ_UP_ARROW, 0, 0, 503);
_expandBtn.addEventListener(MouseEvent.CLICK, onSelectionClick, false, 0, true);
_expandBtn.visible = false;


_hitArea = new Sprite();
resizeHitArea();

addChild(_bg);
addChild(_hitArea);
addChild(_expandBtn);
addChild(_contractBtn);
addChild(_selectionName);
addChild(_selectionInfo);

layout();
}

private function layout():void
{
_selectionName.y = 10;
_selectionName.x = 6;

_selectionInfo.y = 40;
_selectionInfo.x = 8;

_contractBtn.y = _selectionName.y + (_selectionName.textHeight - _contractBtn.height) * 0.5
_expandBtn.y = _selectionName.y + (_selectionName.textHeight - _expandBtn.height) * 0.5
}

private function resizeHitArea():void
{
_hitArea.graphics.clear();
_hitArea.graphics.beginFill(0x000000, 0.001);
_hitArea.graphics.drawRect(0, 0, _scrollRect.width, _scrollRect.height);
_hitArea.graphics.endFill();
_hitArea.mouseEnabled = false;

_contractBtn.hitArea
```

```
= _hitArea;
_expandBtn.hitArea = _hitArea;
}

public function setInfoText( text:String, color:uint ):void
{
_selectionInfo.textColor = color;
_selectionInfo.htmlText = text;
}

public function get infoText():String
{
return _selectionInfo.text;
}

public function set frameName( v:String ):void
{
_selectionName.text = v;
layout();
}

public function set selection( v:* ):void
{
_selection = v;
}

public function get selection():*
{
return _selection;
}

public function set extended( v:Boolean ):void
{
_extended = v;

_expandBtn.visible = _extended;
_contractBtn.visible = !_extended;

TweenLite.killTweensOf(_bg);
TweenLite.killTweensOf(_scrollRect);

var height:int = (_extended) ? (_selectionInfo.y + _selectionInfo.height + 10) : 43;
var easeFunction:Function = (_extended) ? Quad.easeOut : Quad.easeIn;
var time:Number = (_extended) ? 0.4 : 0.3;

TweenLite.to(_bg, time, {height:height, ease:easeFunction, onComplete:onResizeComplete});
TweenLite.to(_scrollRect, time, {height:(height + 1), ease:easeFunction,
onUpdate:onScrollRectUpdate});

}

public
```

```
function get extended():Boolean
{
return _extended;
}

public function get sort():Number
{
return _sort;
}

public function set sort( v:Number ):void
{
_sort = v;
}

public function get filter():String
{
return _filter;
}

public function set filter( v:String ):void
{
_filter = v;
}

override public function get height():Number
{
return _bg.height;
}

override public function get width():Number
{
return _bg.width;
}

override public function set width( value:Number ):void
{
_bg.width = value;
}

override public function set height( value:Number ):void
{
_bg.height = value;
}

public function get selectionName():String
{
return _selectionName.text;
}

private
```

```
function onScrollRectUpdate():void
{
this.scrollRect = _scrollRect;
onSizeUpdated.dispatch();
}

private function onResizeComplete():void
{
_scrollRect.height = _bg.height + 1;
this.scrollRect = _scrollRect;
resizeHitArea();
onResizeFinish.dispatch(this);
}

private function onSelectionClick( e:MouseEvent ):void
{
onClicked.dispatch(this);
}

public function destroy():void
{
_bg = null;
_hitArea = null;
_scrollRect = null;
_selection = null;

if (_contractBtn)
{
_contractBtn.removeEventListener(MouseEvent.CLICK, onSelectionClick);
_contractBtn.destroy();
}

_contractBtn = null;

if (_expandBtn)
{
_expandBtn.removeEventListener(MouseEvent.CLICK, onSelectionClick);
_expandBtn.destroy();
}

_expandBtn = null;

if (_selectionName)
_selectionName.destroy();

_selectionName = null;

if (_selectionInfo)
_selectionInfo.destroy();

_selectionInfo
```

```
= null;

if (onClicked)
onClicked.removeAll();

onClicked = null;

if (onSizeUpdated)
onSizeUpdated.removeAll();

onSizeUpdated = null;
}
}
}

-----------------------------------
File 825: igw\com\ui\modal\intro\FAQView.as
package com.ui.modal.intro
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.enum.ui.PanelEnum;
import com.model.asset.AssetVO;
import com.model.prototype.IPrototype;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.ScaleBitmap;
import com.ui.core.View;
import com.ui.core.component.accordian.AccordianComponent;
import com.ui.core.component.accordian.AccordianGroup;
import com.ui.core.component.bar.VScrollbar;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.filterlist.FilterList;
import com.ui.core.component.label.Label;
import com.ui.core.component.tooltips.Tooltips;
import com.ui.modal.ButtonFactory;

import flash.display.Bitmap;
import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.geom.Rectangle;
import flash.text.TextFormatAlign;
import flash.utils.Dictionary;
import flash.utils.getDefinitionByName;

import org.shared.ObjectPool;

public class FAQView extends View
{
```

```
private var _bg:DefaultWindowBG;

private var _accordian:AccordianComponent;

private var _groupID:String;
private var _maxHeight:int;

protected var _scrollRect:Rectangle;

private var _container:Sprite;
private var _selectionHolder:Sprite;

private var _eightsImage:Bitmap;

private var _scrollbar:VScrollbar;

private var _componentPanelSelections:Dictionary;
private var _visibleComponentPanelSelections:Vector.<FAQSelectionComponent>;

private var _selectedComponent:FAQSelectionComponent;

private var _titleText:String = 'CodeString.FAQView.Title'; //ASK EIGHTS

[PostConstruct]
override public function init():void
{

super.init();

_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(736, 486);
_bg.addTitle(_titleText, 240);
addListener(_bg.closeButton, MouseEvent.CLICK, onClose);

_eightsImage = UIFactory.getBitmap("AskEightsBMD");
_eightsImage.x = 28;
_eightsImage.y = 8;

_accordian = ObjectPool.get(AccordianComponent);
_accordian.init(151, 30);
_accordian.x = _eightsImage.x;
_accordian.y = _eightsImage.y + _eightsImage.height;
_accordian.addListener(onAccordianSelected);

_container = UIFactory.getHeaderPanel(PanelEnum.CONTAINER_NOTCHED,
PanelEnum.HEADER_NOTCHED_RIGHT, 551, 439, 30, _eightsImage.x + _eightsImage.width,
50, "MEH", LabelEnum.
H3);

_componentPanelSelections
```

```
= new Dictionary;
_visibleComponentPanelSelections = new Vector.<FAQSelectionComponent>;

_selectionHolder = new Sprite();
_selectionHolder.x = _eightsImage.x + _eightsImage.width + 6;
_selectionHolder.y = _bg.y + 85;
_maxHeight = 0;

_scrollRect = new Rectangle(0, _bg.y + 23, _bg.width - 20, 428);
_scrollRect.y = 0;
_selectionHolder.scrollRect = _scrollRect

_scrollbar = new VScrollbar();
var dragBarBGRect:Rectangle = new Rectangle(0, 5, 5, 2);
_scrollbar.init(7, _scrollRect.height - 8, _container.x + _container.width - 31, _container.y + 34,
dragBarBGRect, '', 'ScrollBarBMD', '', false, this);
_scrollbar.onScrollSignal.add(onChangedScroll);
_scrollbar.updateScrollableHeight(_maxHeight);
_scrollbar.updateDisplayedHeight(_scrollRect.height);
_scrollbar.maxScroll = 10.75;

addChild(_bg);
addChild(_accordian);
addChild(_container);
addChild(_selectionHolder);
addChild(_scrollbar);
addChild(_eightsImage);

var options:Vector.<IPrototype> = new Vector.<IPrototype>;
var prototypes:Vector.<IPrototype> = presenter.getFAQPrototypes();
for (var i:int = 0; i < prototypes.length; i++)
options.push(prototypes[i]);

addSelections(options);

addEffects();
effectsIN();
}

private function onAccordianSelected( groupID:String, subItemID:String, data:* ):void
{
_groupID = groupID;

var group:AccordianGroup = _accordian.getGroup(groupID);
setContainerTitle(group.text);

if (_selectedComponent)
{
_selectedComponent.extended = !_selectedComponent.extended;
_selectedComponent = null;
}
```

```
if (_selectionHolder.numChildren > 0)
_selectionHolder.removeChildren(0, (_selectionHolder.numChildren - 1));

_visibleComponentPanelSelections.length = 0;
for each (var value:FAQSelectionComponent in _componentPanelSelections)
{
if (value.filter == _groupID)
{
_selectionHolder.addChild(value);
_visibleComponentPanelSelections.push(value);
}
}

_visibleComponentPanelSelections.sort(orderItems);

layout();
_scrollbar.resetScroll();
}

private function onChangedScroll( percent:Number ):void
{
_scrollRect.y = (_maxHeight - _scrollRect.height) * percent;
_selectionHolder.scrollRect = _scrollRect;
}

private function orderFilters( filterOne:AssetVO, filterTwo:AssetVO ):Number
{
if (!filterOne)
return -1;
if (!filterTwo)
return 1;

if (filterOne.sort < filterTwo.sort)
return -1;
else
return 1;
}

private function orderItems( itemOne:FAQSelectionComponent,
itemTwo:FAQSelectionComponent ):Number
{
if (!itemOne)
return -1;
if (!itemTwo)
return 1;

var sortOne:Number = itemOne.sort;
var sortTwo:Number = itemTwo.sort;

var
```

```
itemOneName:String = itemOne.selectionName.toLowerCase();
var itemTwoName:String = itemTwo.selectionName.toLowerCase();

if (sortOne < sortTwo)
return -1;
else if (sortOne > sortTwo)
return 1;

if (itemOneName > itemTwoName)
return 1;
else if (itemOneName < itemTwoName)
return -1;

return 0;
}

private function addSelections( components:Vector.<IPrototype> ):void
{
var currentSelectionComponent:FAQSelectionComponent;
var assetVO:AssetVO;
var filterAssetVO:AssetVO;
var type:String;
var i:uint;
var filters:Array = new Array();
var len:uint = components.length;
var proto:IPrototype;
var image:String;
for (i = 0; i < len; ++i)
{
proto = components[i];
assetVO = presenter.getAssetVOFromIPrototype(proto);
filterAssetVO = presenter.getFilterAssetVO(proto);

if (components[i].name in _componentPanelSelections)
currentSelectionComponent = _componentPanelSelections[components[i].name];
else
currentSelectionComponent = new FAQSelectionComponent();

currentSelectionComponent.setInfoText(assetVO.descriptionText, 0xf0f0f0);
currentSelectionComponent.selection = components[i];
currentSelectionComponent.frameName = assetVO.visibleName;
currentSelectionComponent.filter = filterAssetVO.type;
currentSelectionComponent.sort = components[i].getUnsafeValue('sort');
currentSelectionComponent.onClicked.add(onSelectedComponent);
currentSelectionComponent.onSizeUpdated.add(layout);
currentSelectionComponent.onResizeFinish.add(resizedFinished);
_componentPanelSelections[components[i].name] = currentSelectionComponent;
if (filters.indexOf(filterAssetVO) == -1)
filters.push(filterAssetVO);
}

filters.sort(orderFilters);
```

```
len = filters.length;
var currentAssetVO:AssetVO;
for (i = 0; i < len; ++i)
{
currentAssetVO = filters[i];
_accordian.addGroup(currentAssetVO.type, currentAssetVO.visibleName);
if (_groupID == null || _groupID == '')
onAccordianSelected(currentAssetVO.type, null, null);
}
}

private function onSelectedComponent( selectionBtn:FAQSelectionComponent ):void
{
var oldComponent:FAQSelectionComponent = _selectedComponent;
_selectedComponent = selectionBtn;
_selectionHolder.setChildIndex(selectionBtn, _selectionHolder.numChildren - 1);

if (oldComponent)
oldComponent.extended = !oldComponent.extended;
else
_selectedComponent.extended = !_selectedComponent.extended;
}

private function layout():void
{
var len:uint = _visibleComponentPanelSelections.length;
var selection:FAQSelectionComponent;
var yPos:int = 1;
_maxHeight = 0;
for (var i:uint = 0; i < len; ++i)
{
selection = _visibleComponentPanelSelections[i];
selection.y = yPos;
_maxHeight += selection.height + 5;
yPos += selection.height + 5;
}
_maxHeight -= 4;
_scrollbar.updateScrollableHeight(_maxHeight);
}

private function resizedFinished( selectionBtn:FAQSelectionComponent ):void
{
if (!selectionBtn.extended)
{
if (_selectedComponent != null)
{
if (selectionBtn != _selectedComponent)
_selectedComponent.extended = !_selectedComponent.extended;
else
_selectedComponent
```

```
= null;

layout();
}
} else
{
layout();

if (selectionBtn.y + selectionBtn.height > _scrollRect.height)
_scrollbar.updateScrollY(selectionBtn.y + selectionBtn.height - _scrollRect.height + 3);
}

if (_maxHeight <= _scrollRect.height)
_scrollbar.resetScroll();
}

private function cleanUpSelections():void
{
for (var key:Object in _componentPanelSelections)
{
_componentPanelSelections[key].destroy();
delete _componentPanelSelections[key];
}
_visibleComponentPanelSelections.length = 0;
}

private function setContainerTitle( v:String ):void { Label(_container.getChildAt(2)).text = v; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function get width():Number { return _bg.width; }
override public function get height():Number { return _bg.height; }

override public function destroy():void
{
super.destroy();

cleanUpSelections();

_selectedComponent = null;
_scrollRect = null;
_container = null;
_selectionHolder = null;
_eightsImage = null;
_componentPanelSelections = null;
_visibleComponentPanelSelections = null;

_selectedComponent = null;

if
```

```
(_bg)
ObjectPool.give(_bg);

_bg = null;

if (_accordian)
ObjectPool.give(_accordian);

_accordian = null;

if (_scrollbar)
_scrollbar.destroy();

_scrollbar = null;
}
}
}


------------------------------------
File 826: igw\com\ui\modal\intro\FTETipView.as
package com.ui.modal.intro
{
import com.enum.ui.ButtonEnum;
import com.enum.ui.LabelEnum;
import com.presenter.shared.IUIPresenter;
import com.ui.UIFactory;
import com.ui.core.DefaultWindowBG;
import com.ui.core.View;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.label.Label;
import com.ui.hud.battle.BattleShipSelectionView;
import com.ui.hud.battle.BattleUserView;
import com.ui.hud.shared.ChatView;
import com.ui.hud.shared.IconDrawerView;
import com.ui.hud.shared.MiniMapView;
import com.ui.hud.shared.PlayerView;
import com.ui.hud.shared.bridge.BridgeView;
import com.ui.hud.shared.command.CommandView;
import com.ui.hud.shared.engineering.EngineeringView;
import com.ui.modal.PanelFactory;

import flash.display.Bitmap;
import flash.events.MouseEvent;
import flash.text.TextFormatAlign;

import org.parade.core.ViewEvent;
import org.parade.enum.ViewEnum;
import org.shared.ObjectPool;

public
```

```
class FTETipView extends View
{
private var _bg:DefaultWindowBG;
private var _eightsImage:Bitmap;
private var _skipButton:BitmapButton;
private var _engageButton:BitmapButton;
private var _offset:Number = 0;
private var _selectedComponent:FAQSelectionComponent;

private var _titleText:String = 'CodeString.FTETipView.Title'; //WELCOME TO THE
MAELSTROM!
private var _trainingBtnText:String = 'CodeString.FTETipView.Training'; //TRAINING
private var _playNowBtnText:String = 'CodeString.FTETipView.PlayNow'; //PLAY NOW
private var _tip1Text:String = 'CodeString.FTETipView.Tip1'; //This sector is controlled by your
faction, but beware of attacks by enemy players
private var _tip2Text:String = 'CodeString.FTETipView.Tip2'; //Upgrade your base and research
new technologies to build stronger ships
private var _tip3Text:String = 'CodeString.FTETipView.Tip3'; //The Mission system will guide
you, reward you, and challenge you
private var _tip4Text:String = 'CodeString.FTETipView.Tip4'; //Do not ignore base defenses!
Your protective shield depletes in 7 days
private var _tip5Text:String = 'CodeString.FTETipView.Tip5'; //Make use of the <font
color='#fac569'>Store</font> (top left) to speed-up your upgrades, research, repair, and ship
construction

[PostConstruct]
override public function init():void
{
super.init();
_bg = ObjectPool.get(DefaultWindowBG);
_bg.setBGSize(555, 354);
_bg.x = 286;
_bg.addTitle(_titleText, 314);
addListener(_bg.closeButton, MouseEvent.CLICK, onSkipFTEClick);

_eightsImage = UIFactory.getBitmap('EightsBMD');
_eightsImage.y = -30;

_engageButton = UIFactory.getButton(ButtonEnum.GREEN_A, 240, 40, 607, 400,
_trainingBtnText, LabelEnum.H1);

_skipButton = UIFactory.getButton(ButtonEnum.BLUE_A, 240, 40, 310, 400, _playNowBtnText,
LabelEnum.H1);

addListener(_engageButton, MouseEvent.CLICK, onClose);
addListener(_skipButton, MouseEvent.CLICK, onSkipFTEClick);

addChild(_bg);
addChild(_eightsImage);
addChild(_skipButton);
addChild(_engageButton);
```

```
createBulletPoint(_tip1Text);
createBulletPoint(_tip2Text);
createBulletPoint(_tip3Text);
createBulletPoint(_tip4Text);
createBulletPoint(_tip5Text);

addEffects();
effectsIN();
}

private function createBulletPoint( txt:String ):void
{
var img:Bitmap = PanelFactory.getPanel("FTEBulletBMD");
img.x = 329;
img.y = 89 + _offset;

var label:Label = new Label(16, 0xfffee6, 475, 100, true, 1);
label.constrictTextToSize = false;
label.letterSpacing = .75;
label.multiline = true;
label.align = TextFormatAlign.LEFT;
label.leading = -3;
label.htmlText = txt;
label.x = img.x + 15;
label.y = img.y - 5;

addChild(img)
addChild(label);
_offset += label.textHeight + 13;
}

override protected function onClose( e:MouseEvent = null ):void
{
presenter.fteNextStep();
destroy();
}

private function onSkipFTEClick( e:MouseEvent ):void
{
var event:ViewEvent = new ViewEvent(ViewEvent.UNHIDE_VIEWS);
event.targetClass = [BridgeView, ChatView, IconDrawerView, PlayerView, EngineeringView,
MiniMapView, CommandView];
presenter.dispatch(event);

presenter.fteSkip();
destroy();
}

override
```

```
public function get type():String { return ViewEnum.HOVER; }

override public function get height():Number { return _bg.height + 48; }
override public function get width():Number { return _bg.width + _bg.x; }

[Inject]
public function set presenter( v:IUIPresenter ):void { _presenter = v; }
public function get presenter():IUIPresenter { return IUIPresenter(_presenter); }

override public function destroy():void
{
super.destroy();
_bg = null;
_eightsImage = null;
_engageButton = null;
}
}
}
```

File 827: igw\com\ui\modal\intro\PulseScaleEffect.as

```
package com.ui.modal.intro
{
import flash.display.DisplayObject;
import flash.display.Sprite;

import org.greensock.TweenManual;

public class PulseScaleEffect extends Sprite
{
protected var _obj:DisplayObject;
protected var _end:Number;
protected var _speed:Number;
protected var _start:Number;

public function PulseScaleEffect( obj:DisplayObject, start:Number, end:Number, speed:Number
= 1 )
{
_obj = obj;
_end = end;
_speed = speed;
_start = start;
scaleUp();
}

protected function scaleDown():void
{
TweenManual.to(_obj, _speed, {scaleX:_start, scaleY:_start, onComplete:scaleUp});
}

protected
```

```
function scaleUp():void
{
TweenManual.to(_obj, _speed, {scaleX:_end, scaleY:_end, onComplete:scaleDown});
}

public function destroy():void
{
TweenManual.killTweensOf(_obj);
_obj.scaleX = _obj.scaleY = 1;
_obj = null;
}

}
}
```

File 828: igw\com\ui\modal\intro\characterselect\CharacterSelection.as

```
package com.ui.modal.intro.characterselect
{
import com.model.prototype.IPrototype;
import com.ui.core.component.button.BitmapButton;
import com.ui.core.component.misc.ImageComponent;
import com.ui.modal.ButtonFactory;
import com.ui.modal.PanelFactory;
import com.util.CommonFunctionUtil;

import flash.display.BitmapData;
import flash.display.Sprite;
import flash.events.MouseEvent;
import flash.filters.ColorMatrixFilter;

import org.greensock.TweenLite;
import org.osflash.signals.Signal;
import org.shared.ObjectPool;

public class CharacterSelection extends BitmapButton
{
private var _raceImage:ImageComponent;
private var _largeRaceImage:ImageComponent;
private var _activated:Boolean;
private var _racePrototype:IPrototype;

public var onClicked:Signal;
public var onRollOver:Signal;
public var onLoadImage:Signal;

public function CharacterSelection()
{
onClicked = new Signal(CharacterSelection, Boolean);
onRollOver = new Signal(CharacterSelection);
onLoadImage
```

```
= new Signal(String, Function);

var unselected:BitmapData = PanelFactory.getBitmapData('PortraitFrameUnselectedBMD');
var selected:BitmapData = PanelFactory.getBitmapData('PortraitFrameSelectedBMD');

_raceImage = ObjectPool.get(ImageComponent);
_raceImage.init(53, 53);

addChild(_raceImage);

super.init(unselected, unselected, unselected, unselected, selected);
}

override protected function onMouse( e:MouseEvent ):void
{
super.onMouse(e);
if (mouseEnabled)
{
switch (e.type)
{
case MouseEvent.CLICK:
setUpLargeImage();
onClicked.dispatch(this, true);
break;
case MouseEvent.ROLL_OVER:
setUpLargeImage();
onRollOver.dispatch(this);
break;
}
}
}

override public function set selected( value:Boolean ):void
{
if (value)
onFadeInComplete();
else
TweenLite.killTweensOf(_bitmap);

super.selected = value;
}

private function onFadeOutComplete():void
{
TweenLite.to(_bitmap, 0.75, {alpha:1, onComplete:onFadeInComplete});
}

private function onFadeInComplete():void
{
TweenLite.to(_bitmap, 0.75, {alpha:0.5, onComplete:onFadeOutComplete});
}
```

```
public function setUpLargeImage():void
{
if (_largeRaceImage == null)
{
_largeRaceImage = ObjectPool.get(ImageComponent);
_largeRaceImage.init(300, 300);
onLoadImage.dispatch(_racePrototype.getUnsafeValue('uiAsset'),
_largeRaceImage.onImageLoaded);
}
}

public function onLoadedImage( asset:BitmapData ):void
{
if (_raceImage && _bitmap)
{
_raceImage.onImageLoaded(asset);
_raceImage.x = _bitmap.x + (_bitmap.width - _raceImage.width) * 0.5
_raceImage.y = _bitmap.y + (_bitmap.height - _raceImage.height) * 0.5
}
}

public function set activated( v:Boolean ):void
{
_activated = v;
}

public function get activated():Boolean
{
return _activated;
}

public function set racePrototype( v:IPrototype ):void
{
_racePrototype = v;
}

public function get racePrototype():IPrototype
{
return _racePrototype;
}

public function get image():ImageComponent
{
setUpLargeImage();
return _largeRaceImage;
}

public function get raceName():String
{
```

```
return _racePrototype.name;
}

public function get race():String
{
return _racePrototype.getValue('race');
}

override public function destroy():void
{
if (_bitmap)
TweenLite.killTweensOf(_bitmap);

super.destroy();

onClicked.removeAll();
onClicked = null;

onRollOver.removeAll();
onRollOver = null;

onLoadImage.removeAll();
onLoadImage = null;

if (_raceImage)
{
ObjectPool.give(_raceImage);
_raceImage = null;
}

if (_largeRaceImage)
{
ObjectPool.give(_largeRaceImage);
_largeRaceImage = null;
}
}

}
}

-------------------------------------
File 829: igw\com\ui\modal\intro\characterselect\CharacterSelectView.as
```