

```
package br.com.optimedia.autor.view
{
    import br.com.optimedia.assets.NotificationConstants;
    import br.com.optimedia.assets.vo.PresentationVO;
    import br.com.optimedia.assets.vo.SubjectVO;
    import br.com.optimedia.autor.AutorFacade;
    import br.com.optimedia.autor.model.SubjectManagerProxy;
    import br.com.optimedia.autor.view.components.SubjectManager;

    import flash.events.Event;
    import flash.events.MouseEvent;
    import flash.events.TimerEvent;
    import flash.net.URLRequest;
    import flash.net.navigateToURL;
    import flash.utils.Timer;

    import mx.collections.ArrayCollection;
    import mx.controls.Alert;
    import mx.rpc.events.ResultEvent;

    import org.puremvc.as3.multicore.interfaces.INotification;
    import org.puremvc.as3.multicore.patterns.mediator.Mediator;

    public class SubjectManagerMediator extends Mediator
    {
        public static const NAME:String = 'SubjectManagerMediator';

        private var subjectManagerProxy:SubjectManagerProxy;

        private var getSubjectsTimer:Timer;

        public function SubjectManagerMediator(viewComponent:Object=null)
        {
            super(NAME, viewComponent);
        }

        override public function onRegister():void
        {
            trace(NAME+".onRegister()");

            view.addEventListener( SubjectManager.SAVE_SUBJECT_EVENT, saveSubject );
            view.addEventListener( SubjectManager.SAVE_PRESENTATION_EVENT, savePresentation );
            view.addEventListener( SubjectManager.DELETE_SUBJECT_EVENT, deleteSubject );
            view.addEventListener( SubjectManager.DELETE_PRESENTATION_EVENT, deletePresentation );

            view.addEventListener( SubjectManager.LOCK_UNLOCK_PRESENTATION_EVENT, toggleLock );

            view.newPresentationBtn.addEventListener(MouseEvent.CLICK, getSkins);
            view.slideEditBtn.addEventListener( MouseEvent.CLICK, slideEditBtnClick );

            view.setRoleID( AutorFacade(facade).userRole );
            view.userID = AutorFacade(facade).userID;

            subjectManagerProxy = facade.retrieveProxy( SubjectManagerProxy.NAME ) as
```

```

SubjectManagerProxy;
    subjectManagerProxy.getSubjects();
    view.presentationSkins = subjectManagerProxy.presentationSkins;

    getSubjectsTimer = new Timer(60000, 0);
    getSubjectsTimer.addEventListener(TimerEvent.TIMER, timerEventHandler);
    getSubjectsTimer.start();
}

override public function onRemove():void {

}

public function get view():SubjectManager
{
    return viewComponent as SubjectManager;
}

override public function listNotificationInterests():Array
{
    return [NotificationConstants.GET_SUBJECTS_OK,
            NotificationConstants.SAVE_SUBJECT_OK,
            NotificationConstants.SAVE_PRESENTATION_OK,
            NotificationConstants.LOCK_PRESENTATION_OK,
            NotificationConstants.GET_SKINS_RESULT,
            NotificationConstants.BACK_TO_SUBJECT_MANAGER];
}

override public function handleNotification(note:INotification):void
{
    switch (note.getName())
    {
        case NotificationConstants.GET_SUBJECTS_OK:
            var subjectSelectedIndex:int = view.subjectGrid.selectedIndex;
            var presentationSelectedIndex:int = view.presentationGrid.selectedIndex;
            view.subjectArray = new ArrayCollection(note.getBody() as Array);
            view.subjectGrid.selectedIndex = subjectSelectedIndex;
            view.presentationGrid.selectedIndex = presentationSelectedIndex;
            if( view.presentationGrid.selectedIndex != -1 ) {
                try {
                    if( PresentationVO(view.presentationGrid.selectedItem).section_id !=
0 ) view.slideEditBtn.enabled=false;
                    else view.slideEditBtn.enabled = true;
                }catch(e:Error) {
                    view.presentationGrid.selectedIndex = -1;
                    presentationSelectedIndex = -1;
                }
            }
            break;
        case NotificationConstants.SAVE_SUBJECT_OK:
            Alert.show("Módulo salvo com sucesso.", "OK");
            view.newSubjectPopUp.closeMe();
            break;
        case NotificationConstants.SAVE_PRESENTATION_OK:
    
```

```

        Alert.show("Tema salvo com sucesso.", "OK");
        view.newPresentationPopUp.closeMe();
        break;
    case NotificationConstants.LOCK_PRESENTATION_OK:
        subjectManagerProxy.getSubjects();
        break;
    case NotificationConstants.GET_SKINS_RESULT:
        view.presentationSkins = new ArrayCollection( note.getBody() as Array );
        break;
    case NotificationConstants.BACK_TO_SUBJECT_MANAGER:
        subjectManagerProxy.getSubjects();
        getSubjectsTimer.start();
        break;
    default:
        break;
    }
}

private function saveSubject(event:Event):void {
    subjectManagerProxy.saveSubject( view.newSubjectPopUp.subjectVO );
}

private function savePresentation(event:Event):void {
    subjectManagerProxy.savePresentation( view.newPresentationPopUp.presentationVO );
}

private function deleteSubject(event:Event):void {
    subjectManagerProxy.deleteSubject( view.subjectGrid.selectedItem as SubjectVO );
}

private function deletePresentation(event:Event):void {
    subjectManagerProxy.deletePresentation( view.presentationGrid.selectedItem as
PresentationVO );
}

private function slideEditBtnClick(event:MouseEvent):void {
    if( AutorFacade(facade).userRole != AutorFacade.IS_OBSERVER ) {
        sendNotification( NotificationConstants.BEGIN_PRESENTATION_EDIT, PresentationVO(
view.presentationGrid.selectedItem) );
        subjectManagerProxy.lockPresentation( PresentationVO(
view.presentationGrid.selectedItem).presentation_id, AutorFacade(facade).userID );
        getSubjectsTimer.stop();
    }
    else {
        sendNotification( NotificationConstants.BEGIN_PRESENTATION_EDIT, PresentationVO(
view.presentationGrid.selectedItem) );
        sendNotification( NotificationConstants.DISABLE_SLIDE_EDITION, 0 );
        //subjectManagerProxy.lockPresentation( PresentationVO(
view.presentationGrid.selectedItem).presentation_id, AutorFacade(facade).userID );
        getSubjectsTimer.stop();
    }

    /* if( view.userRole == AutorFacade.IS_OBSERVER ) {
        navigateToURL( new URLRequest(

```

```

'http://www.educar.tv/sinase.moodle/interactive/observer.php?presentation='+PresentationVO(
view.presentationGrid.selectedItem).presentation_id) );
    }
    else {
        if( PresentationVO(view.presentationGrid.selectedItem).locked_by == 0 ||
PresentationVO(view.presentationGrid.selectedItem).locked_by == view.userID ) {
            sendNotification( NotificationConstants.BEGIN_PRESENTATION_EDIT,
PresentationVO(view.presentationGrid.selectedItem) );
            subjectManagerProxy.lockPresentation( PresentationVO(
view.presentationGrid.selectedItem).presentation_id, AutorFacade(facade).userID );
        }
        else {
            sendNotification( NotificationConstants.BEGIN_PRESENTATION_VIEW,
PresentationVO(view.presentationGrid.selectedItem) );
        }
        getSubjectsTimer.stop();
    } */
}

private function getSkins(event:MouseEvent):void {
    subjectManagerProxy.getSkins();
}

private function toggleLock(event:ResultEvent):void {
    var presentation:PresentationVO = event.result as PresentationVO;
    if( presentation.locked_by == 0 ) {
        subjectManagerProxy.lockPresentation( presentation.presentation_id, view.userID
);
    }
    else {
        subjectManagerProxy.unlockPresentation( presentation.presentation_id );
    }
}

private function timerEventHandler(e:TimerEvent):void {
    subjectManagerProxy.getSubjects();
}
}
}
}

```