```
package br.com.optimedia.autor.view
{
    import br.com.optimedia.assets.NotificationConstants;
    import br.com.optimedia.assets.vo.SlideCommentVO;
    import br.com.optimedia.autor.AutorFacade;
    import br.com.optimedia.autor.model.SlideManagerProxy;
    import br.com.optimedia.autor.view.components.CommentList;

    import flash.events.Event;
    import flash.events.MouseEvent;
    import flash.system.System;

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

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

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

        private var slideManagerProxy:SlideManagerProxy;

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

        override public function onRegister():void
        {
            trace(NAME+".onRegister()");
            slideManagerProxy = facade.retrieveProxy( SlideManagerProxy.NAME ) as
SlideManagerProxy;
            slideManagerProxy.getSlideComments( view.slideID );
            view.saveBtn.addEventListener(MouseEvent.CLICK, saveComment);
            view.addEventListener(CloseEvent.CLOSE, closeMe);
            view.addEventListener(CommentList.DELETE_COMMENT_EVENT, deleteComment);
        }

        override public function onRemove():void {
            view.removeMe();
            System.gc();
        }

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

        override public function listNotificationInterests():Array
        {
```

```
            return [NotificationConstants.GET_SLIDE_COMMENTS_RESULT,
                    NotificationConstants.SAVE_SLIDE_COMMENT_RESULT,
                    NotificationConstants.DELETE_SLIDE_COMMENT_RESULT];
        }

        override public function handleNotification(note:INotification):void
        {
            switch (note.getName())
            {
                case NotificationConstants.GET_SLIDE_COMMENTS_RESULT:
                    view.commentListDataProvider = new ArrayCollection( note.getBody() as Array
);
                    if(view.commentListDataProvider.length > 0) {
                        view.noCommentsText.visible = false;
                    }
                    else {
                        view.noCommentsText.visible = true;
                    }
                    break;
                case NotificationConstants.SAVE_SLIDE_COMMENT_RESULT:
                    slideManagerProxy.getSlideComments( view.slideID );
                    view.newCommentTextArea.text = 'Deixe seu comentário';
                    view.newCommentHBox.enabled = true;
                    view.currentState = '';
                    break;
                case NotificationConstants.DELETE_SLIDE_COMMENT_RESULT:
                    slideManagerProxy.getSlideComments( view.slideID );
                    break;
                default:
                    break;
            }
        }

        private function saveComment(e:MouseEvent):void {
            if( view.newCommentTextArea.text == '' || view.newCommentTextArea.text == 'Deixe
seu comentário') {
                Alert.show("É necessário preencher um comentário", "Atenção");
            }
            else {
                view.newCommentHBox.enabled = false;
                var comment:SlideCommentVO = new SlideCommentVO();
                comment.user_id = AutorFacade(facade).userID;
                comment.slide_id = view.slideID;
                comment.body = view.newCommentTextArea.text;
                slideManagerProxy.saveSlideComment( comment );
            }
        }

        private function closeMe(e:Event):void {
            AutorFacade(facade).dispose( view );
        }

        private function deleteComment(event:ResultEvent):void {
            slideManagerProxy.deleteSlideComment( event.result as int );
```

```
            }
        }
}
```