```
package br.com.optimedia.autor.model
{
      import br.com.optimedia.assets.FaultHandler;
      import br.com.optimedia.assets.NotificationConstants;
      import br.com.optimedia.assets.vo.SlideCommentVO;
      import br.com.optimedia.assets.vo.SlideVO;

      import mx.collections.ArrayCollection;
      import mx.controls.Alert;
      import mx.rpc.AsyncToken;
      import mx.rpc.Responder;
      import mx.rpc.events.FaultEvent;
      import mx.rpc.events.ResultEvent;
      import mx.rpc.remoting.mxml.RemoteObject;

      import org.puremvc.as3.multicore.patterns.proxy.Proxy;

      public class SlideManagerProxy extends Proxy
      {
            public static const NAME:String = "SlideManagerProxy";

            private var remoteService:RemoteObject;

            public function SlideManagerProxy(data:Object=null)
            {
                  super(NAME, data);
            }

            override public function onRegister():void {
                  trace(NAME+".onRegister()");
                  remoteService = new RemoteObject();
                  remoteService.destination = "amfphp";
                  remoteService.source = "autor.SlideManager";
                  remoteService.showBusyCursor = false;
            }

            private function generalFault(event:FaultEvent):void {
                  FaultHandler.handleFault(event);
            }

            public function getSlides(presentationID:uint):void {
                  var asynkToken:AsyncToken =
remoteService.getSlides(presentationID);
                  asynkToken.addResponder( new Responder(getSlidesResult,
generalFault) );
            }
            private function getSlidesResult(event:ResultEvent):void {
                  if( event.result is Array ) {
                        sendNotification(
NotificationConstants.GET_SLIDES_OK, event.result );
                  }
            }

            public function addNewSlide(slideVO:SlideVO):void {
                  var asynkToken:AsyncToken = remoteService.saveSlide(
slideVO );
                  asynkToken.addResponder( new
Responder(addNewSlideResult, generalFault) );
```

```
            }
            private function addNewSlideResult(event:ResultEvent):void {
                  if( event.result == false ) {
                        Alert.show("Não foi possível adicionar o slide",
"Erro");
                  }
                  else if( event.result is SlideVO ) {
                        sendNotification(
NotificationConstants.ADD_NEW_SLIDE_RESULT, event.result );
                  }
            }

            public function setOrder(slideArray:ArrayCollection):void {
                  var i:int = 0;
                  for each( var item:SlideVO in slideArray ) {
                        i++;
                        item.page_order = i;
                  }
                  var asynkToken:AsyncToken =
remoteService.setOrder(slideArray);
                  asynkToken.addResponder( new Responder(setOrderResult,
generalFault) );
            }
            private function setOrderResult(event:ResultEvent):void {
                  if( event.result is Array ) {
                        sendNotification(
NotificationConstants.SET_SLIDE_ORDER_RESULT, event.result );
                  }
            }

            public function deleteSlide(slideVO:SlideVO):void {
                  var asynkToken:AsyncToken =
remoteService.deleteSlide(slideVO);
                  asynkToken.addResponder( new
Responder(deleteSlideResult, generalFault) );
            }
            private function deleteSlideResult(event:ResultEvent):void {
                  if( event.result == false ) {
                        Alert.show("Não foi possível remover o slide",
"Erro");
                  }
                  else if( event.result is Array ) {
                        sendNotification(
NotificationConstants.DELETE_SLIDE_RESULT, event.result );
                  }
            }

            public function saveSlide(slideVO:SlideVO):void {
                  var asynkToken:AsyncToken =
remoteService.saveSlide(slideVO);
                  asynkToken.addResponder( new Responder(saveSlideResult,
generalFault) );
            }
            private function saveSlideResult(event:ResultEvent):void {
                  if( event.result is SlideVO ) {
                        sendNotification(
NotificationConstants.SAVE_SLIDE_RESULT, event.result );
                  }
```

```
            }

            public function saveSlideComment( slideComment:SlideCommentVO
):void {
                  var asynkToken:AsyncToken =
remoteService.saveSlideComment( slideComment );
                  asynkToken.addResponder( new
Responder(saveSlideCommentResult, generalFault) );
            }
            private function
saveSlideCommentResult(event:ResultEvent):void {
                  if( event.result == true ) {
                        sendNotification(
NotificationConstants.SAVE_SLIDE_COMMENT_RESULT, event.result );
                  }
            }

            public function getSlideComments( slideID:int ):void {
                  var asynkToken:AsyncToken =
remoteService.getSlideComments( slideID );
                  asynkToken.addResponder( new
Responder(getSlideCommentsResult, generalFault) );
            }
            private function
getSlideCommentsResult(event:ResultEvent):void {
                  if( event.result is Array ) {
                        sendNotification(
NotificationConstants.GET_SLIDE_COMMENTS_RESULT, event.result );
                  }
            }

            public function deleteSlideComment( commentID:int ):void {
                  var asynkToken:AsyncToken =
remoteService.deleteSlideComment( commentID );
                  asynkToken.addResponder( new
Responder(deleteSlideCommentResult, generalFault) );
            }
            private function
deleteSlideCommentResult(event:ResultEvent):void {
                  if( event.result == true ) {
                        sendNotification(
NotificationConstants.DELETE_SLIDE_COMMENT_RESULT, event.result );
                  }
                  else {
                        Alert.show("Não foi possível remover o comentário,
tente novamente", "Erro");
                  }
            }
      }
}
```