```
package br.com.optimedia.autor.view
{
    import br.com.optimedia.assets.NotificationConstants;
    import br.com.optimedia.assets.vo.MediaVO;
    import br.com.optimedia.assets.vo.PresentationVO;
    import br.com.optimedia.autor.model.RepositoryManagerProxy;
    import br.com.optimedia.autor.view.components.RepositoryPanel;

    import flash.events.MouseEvent;

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

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

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

        private var proxy:RepositoryManagerProxy;

        //private var presentationID:uint;

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

        override public function onRegister():void
        {
            trace(NAME+".onRegister()");
            view.deleteBtn.addEventListener( MouseEvent.CLICK, deleteMedia );
            view.linkBtn.addEventListener( MouseEvent.CLICK, doLink);

            proxy = facade.retrieveProxy( RepositoryManagerProxy.NAME ) as
RepositoryManagerProxy;
        }

        override public function onRemove():void {

        }

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

        override public function listNotificationInterests():Array
        {
            return [NotificationConstants.BEGIN_PRESENTATION_EDIT,
                    NotificationConstants.GET_MEDIAS_RESULT,
                    NotificationConstants.DELETE_MEDIA_OK];
        }

        override public function handleNotification(note:INotification):void
        {
            switch (note.getName())
            {
                case NotificationConstants.BEGIN_PRESENTATION_EDIT:
                    view.subjectID = PresentationVO( note.getBody() ).subject_id;
                    proxy.getMedias( view.subjectID );
                    break;
                case NotificationConstants.GET_MEDIAS_RESULT:
                    view.mediaXml = new ArrayCollection(note.getBody() as Array);
                    view.mediaTree.validateNow();
                    //habilite para a tree aparecer toda aberta
                    /* for (var i:int = 0; i < view.mediaXml.length; i ++){
                        view.mediaTree.expandItem(view.mediaXml[i], true)
                    } */
                    break;
                case NotificationConstants.DELETE_MEDIA_OK:
                    proxy.getMedias( view.subjectID );
                    Alert.show("Mídia removida com sucesso", "OK");
                    break;
                default:
                    break;
            }
        }

        private function deleteMedia(event:MouseEvent):void {
            Alert.noLabel = "Não";
            Alert.yesLabel = "Sim";
            Alert.show("Tem certeza que deseja apagar a media "+MediaVO(
view.mediaTree.selectedItem).title+"?", "Atenção",Alert.YES|Alert.NO, null, deleteSubject, null,
 Alert.NO);
            function deleteSubject(event:CloseEvent):void {
                if(event.detail == Alert.YES) {
                    proxy.deleteMedia( MediaVO(view.mediaTree.selectedItem).media_id );
                }
            }
        }

        private function doLink(event:MouseEvent):void {
            event.stopImmediatePropagation();
            event.stopPropagation();
            event.preventDefault();
            sendNotification( NotificationConstants.DO_LINK_EVENT, view.mediaTree.selectedItem );
            view.linkBtn.enabled = false;
            view.mediaTree.selectedIndex = -1;
            view.previewImage.visible = false;
            view.previewTextArea.visible = false;
        }
    }
}
```