

AcroTeX.Net

**AcroTeX PDF Blog**  
**Scripting Bridge**  
**JavaScript to ActionScript**

**D. P. Story**

The source files for this AcroTeX PDF Blog are attached to this PDF.  
Click each of the links to save the files:

- [vidDispJSControls.mxml](#)
- [vidDispJSControls.swf](#)

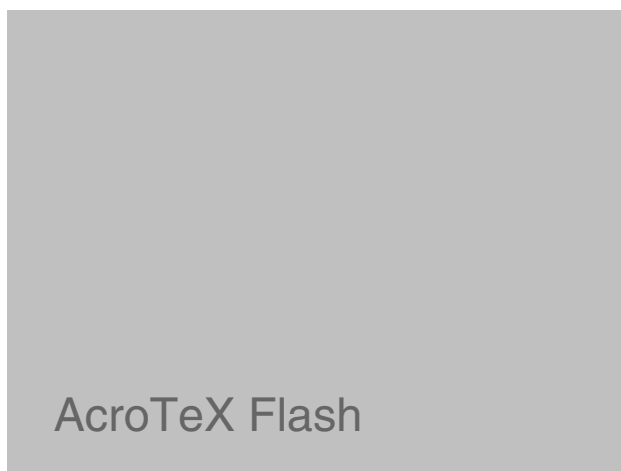
These are the files for the rich media annotation on [page 2](#).

## 1. Introduction

In this article, we discuss the JavaScript to ActionScript communication, a topic that has already been covered in [AcroTeX Blog #1](#). In that article, we used an Acrobat form button to play a Flash video. The button communicated with the built-in player `VideoPlayer.swf` that ships with Acrobat Pro (Extended). Here we use our own video player—one based on the players presented in [AcroTeX Blog #4](#)—and build controls using Acrobat form buttons.

## 2. JavaScript to ActionScript

We begin by illustrating the JavaScript to ActionScript connection link with an example



On the JavaScript side of the infamous scripting bridge, a function that has been exposed by the `ExternalInterface.addCallback` method in ActionScript can be called using the `AnnotRichMedia.callAS` method. The SWF application above exposes three methods that may be called using `callAS`, they are named simply `play`, `pause`, and `stop`.

**Viewing Area JS Actions of Buttons:** Click [play](#), [pause](#), and [stop](#) to view code. ([Clear all](#))

The MXML listing of [vidDispJSControls.mxml](#) is given below, the MXML file itself and the SWF file, [vidDispJSControls.swf](#), are attached to this PDF and may be saved to your local hard drive. The code shows how play, pause, and stop were exposed to the callAS method on the JavaScript side.

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
3   width="100%" height="100%" layout="vertical"
4   horizontalAlign="center" verticalAlign="middle"
5   paddingBottom="0" paddingLeft="0" paddingRight="0" paddingTop="0"
6   creationComplete="initApp()">
7 <mx:Script>
8   <![CDATA[
9     import flash.external.ExternalInterface;
10
11     [Bindable] public var source:String =
12     "http://www.math.uakron.edu/~dpstory/DPS_Demo.flv";
13     private function initApp():void {
14         ExternalInterface.addCallback("play", myVid.play);
15         ExternalInterface.addCallback("pause", myVid.pause);
16         ExternalInterface.addCallback("stop", myVid.stop);
17         ExternalInterface.addCallback("multimedia_play", myVid.play);
18         ExternalInterface.addCallback("multimedia_pause", myVid.pause);
19         ExternalInterface.addCallback("multimedia_stop", myVid.stop);
20         stage.scaleMode = StageScaleMode.EXACT_FIT;
21     }
22   ]]>
23 </mx:Script>
24   <mx:VideoDisplay id="myVid" height="100%" width="100%"
25     source="{source}" autoPlay="true" />
26 </mx:Application>

```

### Comments on the Code

- Line 6: Set the creationComplete property to "initApp()"
- Line 8: Import the ExternalInterface class
- Lines 11–12: Hard-wire the URL to the Flash video
- Lines 13–21: The initApp() function is defined. This function executes during the creationComplete event.
  - Lines 14–19: Add call backs using ExternalInterface.addCallback.<sup>1</sup> The syntax for addCallback is
 

```
addCallback(functionName:String, closure:Function):void
```

<sup>1</sup>[http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/flash/external/ExternalInterface.html#addCallback\(\)](http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/flash/external/ExternalInterface.html#addCallback())

Think of the first argument as the name used externally by JavaScript, and the second argument is the function ActionScript calls when `functionName` is invoked using the `callAS` method from JavaScript.

- Line 20: Set the scale mode to `EXACT_FIT`, see [AcroTeX Blog #2](#) for a discussion of scale modes.
- Lines 24–25: Use the `VideoDisplay` tag to create a video player, set the source as the URL to the Flash video, and put `autoPlay` to `true`. With `autoPlay="true"`, the media will play when the annotation is activated.

Note that `multimedia_play`, `multimedia_pause`, and `multimedia_stop` are exposed as well and are aliases for `play`, `pause`, and `stop`, respectively. The former are the names exposed by `addCallback` in the built-in players `VideoPlayer.swf` and `AudioPlayer.swf`, as described in [AcroTeX Blog #1](#).

That's pretty much it for now, I simply must get back to my retirement. ☹