



Oddcast Tech Note No. 6
Converting emulated AS3 embed to native AS3 embed

A. Introduction

This note is intended to help facilitate the conversion of old-style *non-native* AS3 embed code, to new style *native* AS3 code.

Read on if -

- you embed your Scenes in your own Flash application written in ActionScript 3 (Flash 9 or higher)
- you embedded your SitePal or Studio Scene or Show in your Flash application before April 14 2009.

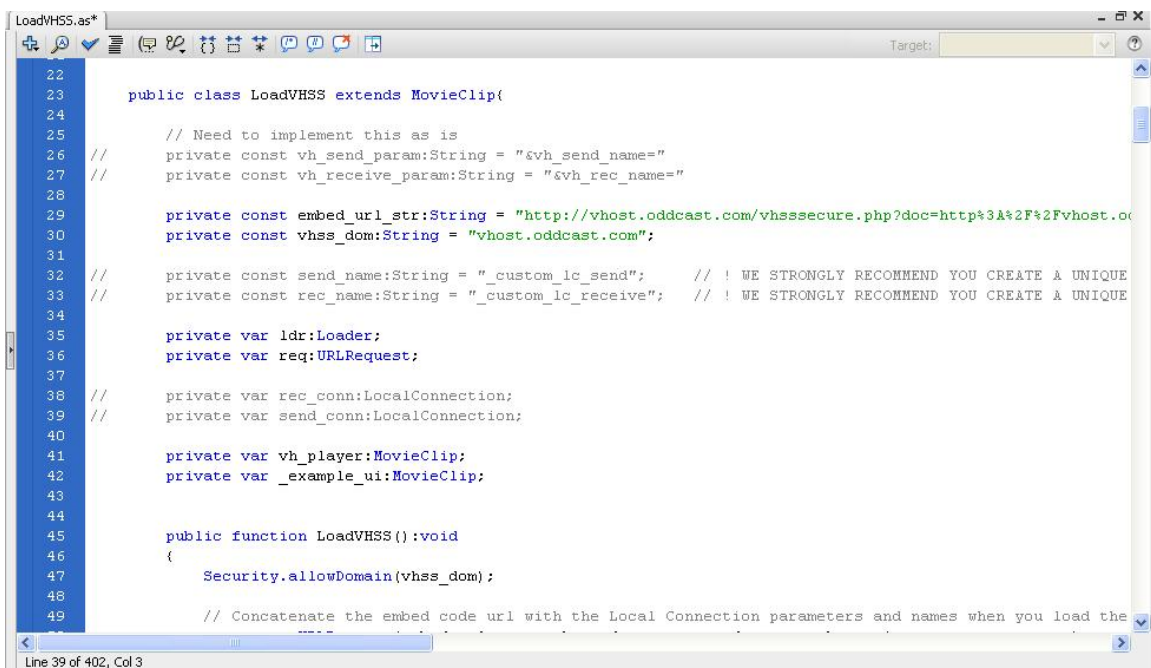
If the above conditions are true, you are likely running an older AS2 version of your character within your AS3 application. Please follow the instructions in this note to update your application to use an AS3 version of the character.

Why should you care -

- Flash performs less optimally when running both AS2 and AS3 code in the same application.
- New features will be made available henceforth only in AS3 characters.
- Oddcast will discontinue support for AS2 character playback at an unspecified time in the future

B. Step by Step Instructions

1. Remove all references to LocalConnection objects and associated parameters. Most importantly, line 50 was previously used to connect the LocalConnection objects to the embed code. This is no longer need and can be simplified as shown to make req equal embed code.

A screenshot of an IDE window titled 'LoadVHSS.as*' showing ActionScript 3 code. The code defines a class 'LoadVHSS' extending 'MovieClip'. It includes several private constants for URLs and parameters, and private variables for 'ldr:Loader', 'req:URLRequest', and 'vh_player:MovieClip'. The 'LoadVHSS()' function is shown starting with 'Security.allowDomain(vhss_dom);'. The code is partially obscured by a blue selection bar on the left side of the editor.

```
22
23     public class LoadVHSS extends MovieClip{
24
25         // Need to implement this as is
26         private const vh_send_param:String = "&vh_send_name="
27         private const vh_receive_param:String = "&vh_rec_name="
28
29         private const embed_url_str:String = "http://vhost.oddcast.com/vhsssecure.php?doc=http%3A%2F%2Fvhost.o
30         private const vhss_dom:String = "vhost.oddcast.com";
31
32         private const send_name:String = "_custom_lc_send";    // ! WE STRONGLY RECOMMEND YOU CREATE A UNIQUE
33         private const rec_name:String = "_custom_lc_receive"; // ! WE STRONGLY RECOMMEND YOU CREATE A UNIQUE
34
35         private var ldr:Loader;
36         private var req:URLRequest;
37
38         private var rec_conn:LocalConnection;
39         private var send_conn:LocalConnection;
40
41         private var vh_player:MovieClip;
42         private var _example_ui:MovieClip;
43
44
45         public function LoadVHSS():void
46         {
47             Security.allowDomain(vhss_dom);
48
49             // Concatenate the embed code url with the Local Connection parameters and names when you load the
```

```

LoadVHSS.as*
42     private var _example_ui:MovieClip;
43
44
45     public function LoadVHSS():void
46     {
47         Security.allowDomain(vhss_dom);
48
49         // Concatenate the embed code url with the Local Connection parameters and names when you load the
50         req = new URLRequest(embed_url_str/* + vh_send_param + send_name + vh_receive_param + rec_name*/);
51
52         /*
53         // Create the sending Local Connection for calls to VHSS api
54         send_conn = new LocalConnection();
55         send_conn.addEventListener(StatusEvent.STATUS, onStatus);
56
57         // Create the receiving Local Connection for VHSS events
58         rec_conn = new LocalConnection();
59         rec_conn.client = this;
60         rec_conn.allowDomain(vhss_dom);
61         rec_conn.allowInsecureDomain(vhss_dom);
62         try {
63             rec_conn.connect(rec_name);
64         } catch (error:ArgumentError) {
65             trace("EXAMPLE --- Can't connect...the connection name is already being used by another SWF");
66         }
67         */
68         configureUI();
69         ldr = new Loader();
70         ldr.addEventListener(Event.COMPLETE, completeHandler);
71         ldr.load(req);
72     }
73 }
Line 66 of 402, Col 3

```

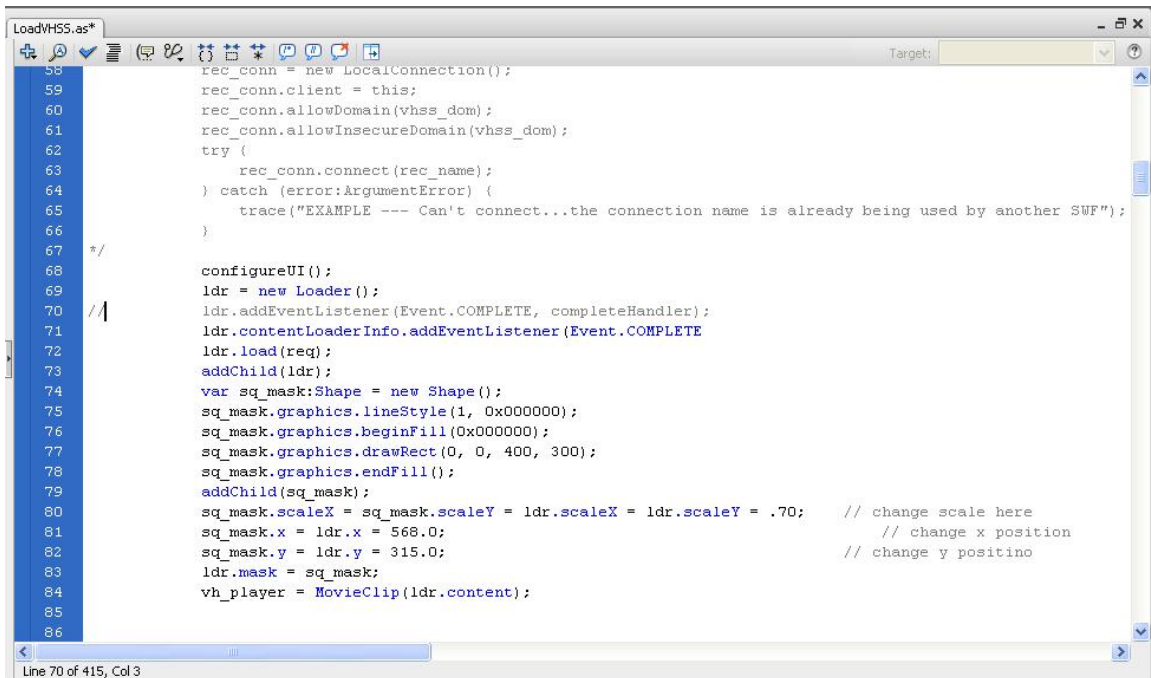
2. Update the embed code and cross domain allowance as shown above.

```

LoadVHSS.as* FlashAPI_A53 fla
23     public class LoadVHSS extends MovieClip{
24
25         // Need to implement this as is
26         private const vh_send_param:String = "&vh_send_name="
27         private const vh_receive_param:String = "&vh_rec_name="
28
29         //
30         private const embed_url_str:String = "http://vhost.oddcast.com/vhsssecure.php?doc=http%3A%2F%2Fvhost.o
31         private const embed_url_str:String = "http://content.oddcast.com/vhss/vhss_v5.swf?doc=http%3A%2F%2Fvhs
32         private const vhss_dom:String = "vhost.oddcast.com";
33         private const vhss_dom:String = "content.oddcast.com";
34
35         //
36         private const send_name:String = "_custom_lc_send"; // ! WE STRONGLY RECOMMEND YOU CREATE A UNIQUE
37         private const rec_name:String = "_custom_lc_receive"; // ! WE STRONGLY RECOMMEND YOU CREATE A UNIQUE
38
39         private var ldr:Loader;
40         private var req:URLRequest;
41
42         //
43         private var rec_conn:LocalConnection;
44         private var send_conn:LocalConnection;
45
46         private var vh_player:MovieClip;
47         private var _example_ui:MovieClip;
48
49         public function LoadVHSS():void
50         {
51             Security.allowDomain(vhss_dom);
52         }
53     }
Line 31 of 406, Col 3

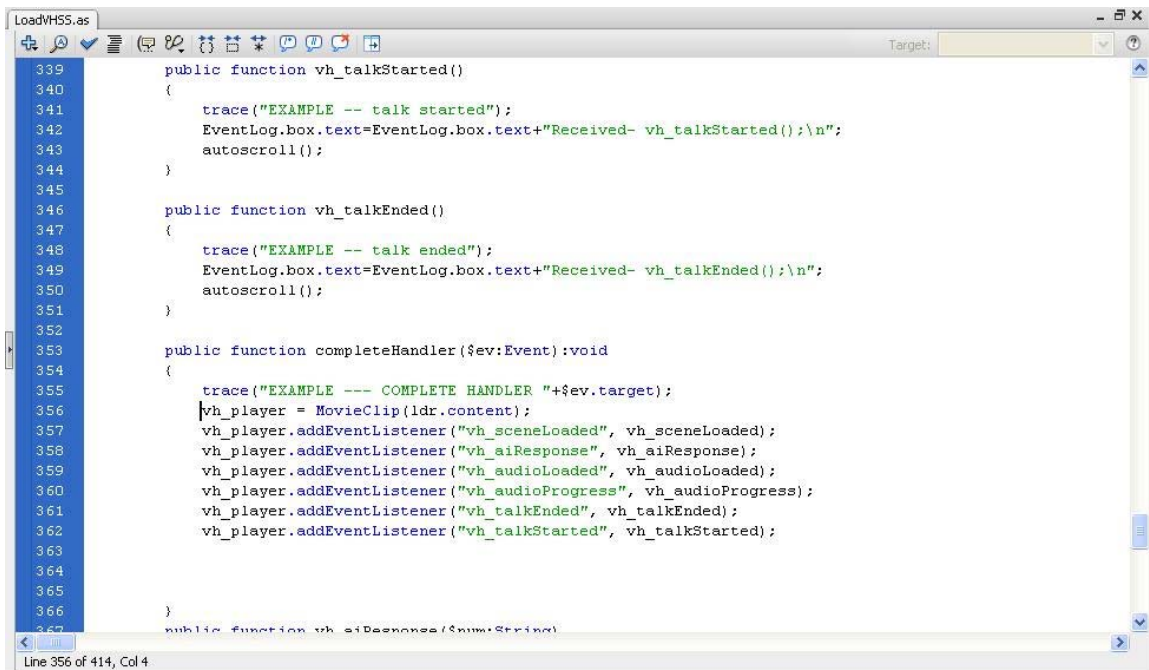
```

3. Update the context of the loader completed event (formerly line 70). In the following screen shot, we commented out the unnecessary line 70 and corrected the context of this call for line 71. This step is very important for extended API use because it allows for registering what player events to receive.



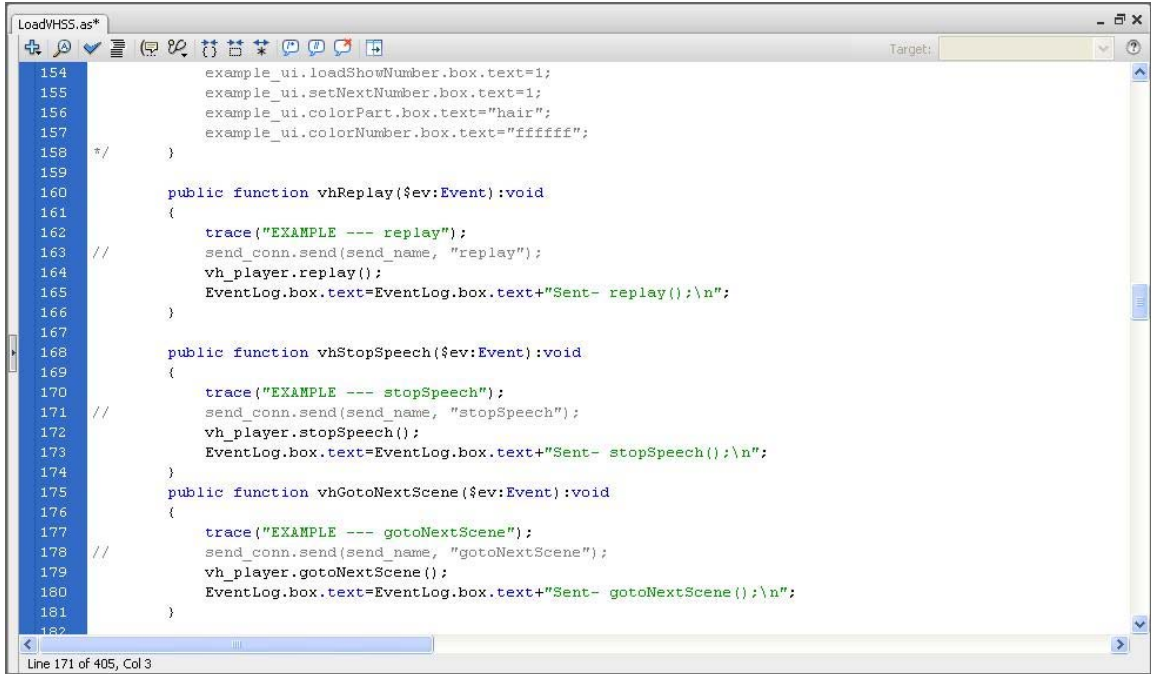
```
LoadVHSS.as
38 rec_conn = new LocalConnection();
59 rec_conn.client = this;
60 rec_conn.allowDomain(vhss_dom);
61 rec_conn.allowInsecureDomain(vhss_dom);
62 try {
63     rec_conn.connect(rec_name);
64 } catch (error:ArgumentError) {
65     trace("EXAMPLE --- Can't connect...the connection name is already being used by another SWF");
66 }
67 */
68
69
70 //
71 ldr.addEventListener(Event.COMPLETE, completeHandler);
72 ldr.contentLoaderInfo.addEventListener(Event.COMPLETE
73 ldr.load(req);
74 addChild(ldr);
75 var sq_mask:Shape = new Shape();
76 sq_mask.graphics.lineStyle(1, 0x000000);
77 sq_mask.graphics.beginFill(0x000000);
78 sq_mask.graphics.drawRect(0, 0, 400, 300);
79 sq_mask.graphics.endFill();
80 addChild(sq_mask);
81 sq_mask.scaleX = sq_mask.scaleY = ldr.scaleX = ldr.scaleY = .70; // change scale here
82 sq_mask.x = ldr.x = 568.0; // change x position
83 sq_mask.y = ldr.y = 315.0; // change y positino
84 ldr.mask = sq_mask;
85 vh_player = MovieClip(ldr.content);
86
```

4. Declare the connection between the ldr.content and the movie clip and any listeners that are required.



```
LoadVHSS.as
339 public function vh_talkStarted()
340 {
341     trace("EXAMPLE -- talk started");
342     EventLog.box.text=EventLog.box.text+"Received- vh_talkStarted();\n";
343     autoscroll();
344 }
345
346 public function vh_talkEnded()
347 {
348     trace("EXAMPLE -- talk ended");
349     EventLog.box.text=EventLog.box.text+"Received- vh_talkEnded();\n";
350     autoscroll();
351 }
352
353 public function completeHandler($ev:Event):void
354 {
355     trace("EXAMPLE --- COMPLETE HANDLER "+$ev.target);
356     vh_player = MovieClip(ldr.content);
357     vh_player.addEventListener("vh_sceneLoaded", vh_sceneLoaded);
358     vh_player.addEventListener("vh_aiResponse", vh_aiResponse);
359     vh_player.addEventListener("vh_audioLoaded", vh_audioLoaded);
360     vh_player.addEventListener("vh_audioProgress", vh_audioProgress);
361     vh_player.addEventListener("vh_talkEnded", vh_talkEnded);
362     vh_player.addEventListener("vh_talkStarted", vh_talkStarted);
363 }
364
365
366
367 public function vh_aiResponse($num:String)
```

5. Change the API calls. The code should now target the movie object itself with the commands. This will be formatted similar to the original AS2 API interactions.



```
LoadVHSS.as*
154     example_ui.loadShowNumber.box.text=1;
155     example_ui.setNextNumber.box.text=1;
156     example_ui.colorPart.box.text="hair";
157     example_ui.colorNumber.box.text="ffffff";
158 */    }
159
160     public function vhReplay($ev:Event):void
161     {
162         trace("EXAMPLE --- replay");
163         // send_conn.send(send_name, "replay");
164         vh_player.replay();
165         EventLog.box.text=EventLog.box.text+"Sent- replay();\n";
166     }
167
168     public function vhStopSpeech($ev:Event):void
169     {
170         trace("EXAMPLE --- stopSpeech");
171         // send_conn.send(send_name, "stopSpeech");
172         vh_player.stopSpeech();
173         EventLog.box.text=EventLog.box.text+"Sent- stopSpeech();\n";
174     }
175     public function vhGotoNextScene($ev:Event):void
176     {
177         trace("EXAMPLE --- gotoNextScene");
178         // send_conn.send(send_name, "gotoNextScene");
179         vh_player.gotoNextScene();
180         EventLog.box.text=EventLog.box.text+"Sent- gotoNextScene();\n";
181     }
182
Line 171 of 405, Col 3
```

If you have any trouble with your conversion, you can contact us at support@oddcast.com

C. Updated AS3 Example

The Posted AS3 embed example has been updated to reflect our Native AS3 embedding and can be found at http://www.oddcast.com/support/as3/VHSS_API_AS3_native.html