Adobe Flash Professional CS5-5.5 Troubleshooting
Legal notices

A note to our customers
Adobe provides this searchable PDF of archived technical support documents as a service to our customers who own and continue to enjoy older, unsupported versions of our software products. The information in these documents is not updated, and will become increasingly less accurate as hardware, browsers, and operating systems continue to evolve. Please be aware that these archived documents reflect historical issues and solutions for products that are no longer supported. Adobe does not warrant that the information in this document is accurate.
Contents

Using Flash to add sound to a Web page: an introduction ........................................... 1
Troubleshooting "Sample: Saving to MS Access using ASP" in Firefly tutorial ........................ 2
Transparent background in a SWF file ........................................................................ 4
Tips for using Flash sprites in Director ........................................................................... 5
Tips for optimizing ActionScript in Adobe Flash movies ............................................. 7
Test Movie, Welcome Screen don’t function correctly | CSS with FileVault enabled .............. 9
Supported file formats | Flash Professional CS5.5 ................................................................. 10
Slow opening transition when editing a symbol in Flash on Windows Vista .................... 12
Set variables in a Flash movie ....................................................................................... 13
Re-create preferences ................................................................................................... 16
Problems using Adobe Flash authoring across local area networks .................................. 20
Pause SWF file | Specific number of seconds .................................................................. 21
How to “park” a movie clip so it can be preloaded and reused ....................................... 22
OBJECT tag syntax | Flash Professional ........................................................................... 23
Multiple instances of movie clip with armature trace only one armature ......................... 25
MovieClip coordinates off on first frame of 3D object | QuickTime (.mov) file | Flash Professional CS5.5 ................................................................. 27
Manually remove Flash Professional CS5.5, CS5, CS4 .................................................. 27
How to make a simple clock in Flash .............................................................................. 33
Leading ampersand breaks text file parsing in ActionScript 3.0 .................................... 34
Large Flash applications compile incorrectly | Flash CS3, CS4, CS5, CS5.5 ...................... 35
Known issues | Flash Professional CS6 .............................................................................. 37
Known issues | Flash CS5.5 ................................................................................................... 38
Fonts rendered incorrectly | Flash | Mac OSX 10.6.7 ......................................................... 40
Adobe Flash Publish keyboard shortcut (Shift+F12) is used by Mac OS X 10.4 ................. 40
Flash Professional CS5 known issues and bugs ............................................................ 41
Security Update 11.5.2 | CS5.5 ......................................................................................... 41
Flash Player 10, Action Script 3: Loader.unloadAndStop ............................................... 42
Flash OBJECT and EMBED tag attributes ................................................................... 43
Flash keyboard shortcuts ............................................................................................... 48
Error "Unexpected element/attribute" | Publish iOS application | Flash Professional CS5.5 ......................................................................................... 50
Error "Could not find an Android device..." | Publish Android app | Flash CS5, CS5.5 ......... 51
How to duplicate a movie clip in random locations ....................................................... 53
Differences between the Equality operator (==) and the Assignment operator (=) .......... 55
Creating a mask that follows a motion guide .................................................................... 56
Creating advanced buttons .............................................................................................. 56
Create a simple sound toggle | Flash ............................................................................. 58
Create pop-up menus | Flash .......................................................................................... 59
How to create password verification .............................................................................. 64
How to create a color fade effect in Flash ...................................................................... 66
Create buttons | Flash .................................................................................................. 66
Child display objects inaccessible after frame navigation | ActionScript 3.0 ...................... 69
<table>
<thead>
<tr>
<th>Characters missing from text in Flash CS4 file opened in Flash CS5</th>
<th>71</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can't install Android app</td>
<td>Flash CS5 CS5.5</td>
</tr>
<tr>
<td>Can't install an AIR for Android app to offline device</td>
<td>Flash CS5, CS5.5</td>
</tr>
<tr>
<td>Cannot type GB18030 characters in panels and dialog boxes</td>
<td>73</td>
</tr>
<tr>
<td>Calling Flash ActionScript functions using Lingo</td>
<td>74</td>
</tr>
<tr>
<td>Bezier Pen Tool</td>
<td>Flash Professional CS</td>
</tr>
<tr>
<td>Adding actions to shared buttons</td>
<td>78</td>
</tr>
</tbody>
</table>
Using Flash to add sound to a Web page: an introduction

**Playing audio on Web pages** Sound files, which come in a variety of formats, are handled differently and inconsistently by different browsers and on different computers. For some measure of predictability and control, users can bring audio into Dreamweaver by first inserting it into a SWF file (created by Macromedia Flash). The SWF file will be played with the Flash plug-in, which comes with most browsers. Flash is a streaming technology, so audio files can start playing even before the entire file has been completely downloaded.

**JavaScript control of audio embedded in an SWF file** A SWF file can be controlled with JavaScript behaviors. For instance, the clicking of a button can trigger the playing of a SWF file which houses an audio file. Unfortunately, a few browsers do not allow JavaScript to control embedded elements such as Flash movies. (See Not all browsers support JavaScript control of embedded content [TechNote 15431] for more information.)

Below is a chart showing which browsers support JavaScript control of embedded elements:

<table>
<thead>
<tr>
<th>Supported browsers</th>
<th>Windows</th>
<th>Macintosh</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Explorer 4.x</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Internet Explorer 5.x</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Netscape 4.x</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Netscape 6</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

**Bringing an MP3 audio file into Dreamweaver using a Flash SWF file** The tutorials in this series use both Flash 5 and Dreamweaver 4 to add audio to a Web page. They each use one kind of audio, MP3 (Motion Picture Experts Group Audio), which is embedded in a SWF file in Flash and then played or controlled in Dreamweaver. Other audio formats (such as WAV for Windows or AIFF for Macintosh) can also be added to the SWF file. The tutorials include:

- Using Flash to add background music to a Web page (TechNote 15345) After embedding an MP3 audio file into a Flash SWF file, you can then add the SWF file to a Dreamweaver document for background sound.

- Using Flash to play audio when a button is clicked (TechNote 15326) Trigger the playing of the SWF file by attaching a JavaScript control behavior to a button or linked text.

- Using Flash to play audio in a specific frame on the timeline (TechNote 15347) Delay the start of the SWF file by placing a behavior on the Dreamweaver timeline.

**Additional information** For more information about working with audio in Flash, see the following Flash TechNotes and articles:
For more information about working with audio in Dreamweaver, see the following Dreamweaver TechNotes and articles:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Link/Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to create a simple sound toggle</td>
<td>TechNote 14821</td>
</tr>
<tr>
<td>Supported audio import formats</td>
<td>TechNote 14068</td>
</tr>
<tr>
<td>Streaming and file optimization techniques for Flash Player</td>
<td></td>
</tr>
</tbody>
</table>

For more information about working with audio in Dreamweaver, see the following Dreamweaver TechNotes and articles:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Link/Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding sound and video to your Web site</td>
<td></td>
</tr>
<tr>
<td>Troubleshooting the Dreamweaver 2 Control Sound behavior</td>
<td>TechNote 14049</td>
</tr>
<tr>
<td>What can prevent JavaScript control over audio?</td>
<td>TechNote 14285</td>
</tr>
<tr>
<td>Playing a sound completely before opening a page</td>
<td>TechNote 12903</td>
</tr>
</tbody>
</table>

**Troubleshooting "Sample: Saving to MS Access using ASP" in Firefly tutorial**

This TechNote covers some of the common pitfalls users come across when working through the Firefly tutorial: Firefly Components Help > Tutorials > Advanced Topics > Sample: Saving to MS Access using ASP.

These tips may also help users troubleshoot errors in their own Firefly applications.

**Connector failed to load data:**


This error occurs when running Control > Test Movie on the sample.fla file. No data appears in the grid and the Output window displays the error message above.

Some possible causes of this error are:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The Virtual Directory named fireflySamples was not created in IIS.</td>
</tr>
<tr>
<td></td>
<td>The Virtual Directory name was spelled incorrectly.</td>
</tr>
<tr>
<td></td>
<td>If you are working with your own Firefly application, check the spelling of the URL to your ASP file in the following places:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FxXMLConnector visual property editor &gt; Properties tab &gt; Source URL field</td>
</tr>
<tr>
<td></td>
<td>FxXMLResolver visual property editor &gt; Properties tab &gt; Resolve URL field</td>
</tr>
</tbody>
</table>

**Resolver couldn’t apply Delta Packet:**

Last updated 11/6/2015
This error occurs when running Control >Test Movie on the sample.fla file. When the user clicks the Add button, enters some information in the new row and then clicks the Save button the Output window displays the error message above. The most probable cause of this error is that the user entered a Record ID number that already exists for another record. The Access databasesample.mdb has the MemberID field defined as a primary key which does not allow duplicate values.

**Error opening Delta Packet::**

This error occurs when running Control >Test Movie on the sample.fla file. When the user clicks the Add button, enters some information in the new row and then clicks the Show Delta Packet button, the Output window displays the error message above. The most probable cause of this error is that the C:\Program Files\Macromedia\Flash MX\Samples\Firefly\ folder does not have the correct Windows permissions. To set the permissions for this folder in Windows 2000:

1. Open Windows Explorer and browse to: C:\Program Files\Macromedia\Flash MX\Samples\Firefly\.
2. Right-click the Firefly folder and choose Properties.
3. Select the Security tab and click the Add button.
4. Choose your local machine from the Look in: field and scroll down to the name IUSR_yourComputer.
5. Click Add and then give the IUSR account Full Control.

**Note:** Windows NT and Windows XP have similar steps. For more details on Windows permissions, please see the following Dreamweaver TechNotes:

* Understanding anonymous authentication and the IUSR account (TechNote 15378)
* Setting IIS web server permissions (TechNote 15376)
* Understanding NTFS permissions (TechNote 15545)

**Delta Packet is empty:**

This issue occurs when running Control > Test Movie on the sample.fla file. When the user clicks the Add button, enters some information in the new row and then clicks the Show Delta Packet button, the Output window just displays the following: <delta_packet/>. This occurs because the information entered in the new row has not yet posted to the dataset. The data is not posted until the user moves to a different record in the grid. Note that the saveUpdates() method automatically calls post before the DeltaPacket is saved to the server. Therefore, an insert would be saved to the server even if the user did not select another record in the grid.
Transparent background in a SWF file

The background color (Stage color) of a SWF file can be set to transparent. The background color or image of the HTML page that contains the SWF file shows through. This technique allows layering of SWF content with DHTML (Dynamic HTML) content.

Not every web browser handles transparency in the same way. Be sure to test your SWF file in all browsers that you want to enable your audience to use. Most Linux browsers do not support Flash transparency. This table lists several browsers that support transparency.

Using the Publish Settings in Flash Professional

The HTML for a SWF file can be created using the Publish Settings feature in Flash. The Publish Settings dialog box provides an option to affect the WMODE setting. The options selected in the Publish Settings are added to the HTML source code automatically.

1. Choose File > Publish Settings. Make sure that HTML is selected.
2. Select HTML.
3. Choose Transparent Windowless from the Window Mode menu to make the SWF file's background disappear in browsers that support this feature.
4. Publish the document.

This video shows how to perform these steps in Flash Professional:

Using the Properties panel in Dreamweaver

Follow the steps below, and Dreamweaver inserts the correct HTML code automatically.

1. In Dreamweaver, insert the SWF file into an HTML page.
2. Select the SWF file in the Design View.
3. In the Properties panel, choose Parameters.
4. For the Parameter, enter "wmode" (without quotes). For the Value, enter "transparent."
5. Save the document. The HTML page is complete.

This video shows how to perform these steps in Dreamweaver:

Editing the HTML code manually

To edit an existing HTML page, add the WMODE parameters to the HTML code.

Add the following parameter to the OBJECT tag:

```html
<param name="wmode" value="transparent"/>
```

For more information about editing HTML tags manually for SWF content, see Object Tag Syntax.
More Help topics
Flash content displays on top of all DHTML layers

Tips for using Flash sprites in Director

Introduction

The following tips and strategies are not all inclusive. The purpose of this TechNote is to address some of the most frequent questions regarding the use of Flash sprites in Director. The material is current for Macromedia Director MX 2004 and Macromedia Flash MX 2004 (SWF 7.0), and the following strategies should be useful for future versions as well.

General approach

It is important to note that it is easier to use Flash sprites if they have been designed specifically for the Director piece. You may encounter difficulties when attempting to incorporate a finished SWF that was originally designed for a browser. Flash files may or may not make a successful translation when being hosted by another application. Although the majority of SWF files may work in Director, not all SWF files can be used in all hosts with equal success.

Scenario 1

A common method for a SWF file designed for browsers utilizes Load Movie actions to link different SWF files together. In Director, the SWF files would be imported as multiple SWF sprites and then controlled through Lingo. If you wish to repurpose a finished SWF file that loads other SWF files, you will modify the original Flash FLA file to create appropriate elements for the new Director project.

Scenario 2:

Flash movies may have originally been designed using the FSCommand actions to speak to the surrounding browser. Although it is possible to write Lingo to accommodate such messages, it is easier to send an actual Lingo: message to the Director host. The choice to keep elements within the control of the target host is also useful for triggering URLs.

For example, when incorporating pre-existing Flash files which use a GetURL action, this script is embedded within Shockwave, which is then embedded within a browser. It is more efficient to design the Flash sprite to pass a message to the Director host, asking that it request an URL from the Shockwave host with the normal goToNetPage command. If command sequences are deeply embedded within each other, troubleshooting is more complex than if each element passes a simple message to its own host.

For additional information, refer to Triggering events from a Flash sprite using "Lingo:" (TechNote 16165). In both of these situations, it may be possible to retrofit a movie that was designed for another host. However, it is usually much easier if the Flash movie was created specifically as media to be hosted in Director.

The Flash Asset Xtra

When creating Flash Sprites or creating Flash objects usingnewObject in Director MX 2004, the Flash Asset Xtra is required. The Flash Asset Xtra provides all the Flash functionality in Director, and the Flash Player does not need to be installed on the end-user’s machine. If the Xtra is not included with the projector, the Director application will not play correctly. For more information about including Xtras in projectors and Shockwave movies, refer to How are Xtras included in Director projectors loaded? (TechNote 12057).

Performance Tips

Last updated 11/6/2015
When authoring Director projects that incorporate Flash assets, developers should follow some specific performance tips. The tips listed under the "Flash > Performance Tips" section of the Director help documentation are essential for optimum playback. The information on the static and frame rate properties are especially useful for increasing performance with some Flash pieces.

Note that when using the Flash Asset Xtra in a projector or Shockwave movie, the CPU usage meter may show 100% usage. This reading does not effect system or movie performance. For more details on why this happens refer to Why does the CPU usage monitor go to 100% when I run Director? (TechNote 8151).

When using Flash Sprites in Director, each Flash Sprite creates a new instance of the Flash Asset Xtra. This means that as more Flash Sprites are used, more memory is required by the application as the computations required exponentially increase. Developers have reported performance difficulties with over twenty simultaneous Flash sprites, which is to be expected. Each overlaid element is turning curves-to-antialiased-pixels many times per second. Real time rendering requires more computations than compositing bitmaps, and this should be taken into consideration when authoring a movie. For more details, please see Why can Shockwave offer higher framerates than Flash? (TechNote 13981) and Multiple Flash assets in Director movies may cause memory problems in projectors (TechNote 15624).

One strategy involves constructing an interface using as many Flash sprites as desired. Once the final layout has been designed, performance is optimized by collapsing multiple SWFs into a single SWF sprite. If all the elements are combined in Flash to create a single SWF (particularly if it can be set as a static SWF), then the computations the processor needs to perform each second will be greatly reduced. The summary for this section is to keep a flattened hierarchy, use simple elements, all at the same level in Director whenever possible. Although it is possible to nest the logic or media deeply within SWFs, it simplifies the authoring process to maintain a set of simple SWFs and allow Lingo to access all the command functions.

**Performance in Director MX 2004**

Director MX 2004 addresses the memory problems associated with multiple Flash assets through the Flash asset commonPlayer. The commonPlayer is a property that applies to all Flash assets, vector shapes, and Flash components. It lets you load multiple Flash sprites into one instance of the Flash Player rather than requiring one Flash Player for each Flash sprite on the stage. The commonPlayer feature is designed to provide better Flash playback performance in Director projects that use large numbers of Flash assets.

**Sound in Flash**

Audio is a special case. It is possible to embed audio in the SWF, inside Director and inside other applications (for example: browsers). If it is necessary to change the audio synch or interactivity, then revisions will require tracing back through multiple files. It simplifies the process if all media elements are accessible within the Director interface, rather than embedded media elements one within another. A Flash button can send a message to the Director host requesting an audio click. This can also resolve difficulties with audio-blending in certain versions. Be aware that some compression options in Flash can conflict with blending of other audio sources on certain hardware systems. For example, the streamsynch property can cause problems which are documented in Flash audio stutters in Director on a Windows system (TechNote 14814). If all the audio is inside Director, the sound files are easier to blend and edit.

**New functionality in Director MX and Director MX 2004**

Director MX now has the ability to access objects within a Flash 6 SWF file. This functionality allows Director to retrieve and set data with Flash native objects and access Flash SWF functionality such as XML parsing, Macromedia Flash Communication Server, Flash Remoting, local and remote shared objects. For more information on this functionality please refer to the Director MX help files.
Macromedia Flash MX 2004 components are bundled movie clips with ActionScript programming interfaces. Director comes with a set of user interface components, including list boxes, radio buttons, check boxes, a scroll pane, and more. Developers can access Flash function directly by simply using:Sprite("flashSprite").FlashFunctionName. For more information see Calling Flash ActionScript functions using Lingo (TechNote 18541). For more information on Cross product support and methodology see the Cross-Product Integration section in the Developers Center.

**Flash in Shockwave issues**

It is important to verify whether or not the Shockwave Player can support the desired Flash features when viewing Flash sprites in Shockwave through browsers. For a list of which versions of Flash are compatible with each version of Director see Flash support in Director (TechNote 14754).

**Summary**

It is possible to use a range of Flash movies inside a Director piece, but it is also possible to create Flash movies which are difficult to use in Director. Keep in mind the following:

- It is generally easiest if the Flash movies used were created specifically to be hosted in Director.
- Use a flattened hierarchy where logic and media are handled in Director rather than embedded in the Flash sprite.
- A good object-oriented design with firm boundaries to individual objects can provide a flexible project with the desired level of performance.

**Tips for optimizing ActionScript in Adobe Flash movies**

If speed is a primary concern, look at the visual aspects of the site. In most cases the majority of speed bottlenecks occur when attempting too many (or too complicated rendering tasks.) Optimizing the visual can achieve significantly greater gains than a slight improvement in code processing.

For optimization tips, see Streaming and file optimization techniques for Flash Player.

Some items to consider:
When possible, try not to manipulate long strings. Break the strings up into small pieces, run the string operations and concatenate the result into a final string for output. The smaller the string the faster the operation will complete. This should also be a consideration with incoming XML data. Large amounts of XML should be broken up before loading into Macromedia Flash if possible.

If manipulating a lot of data on the client, consider using the server to pre-filter the data that the client actually sees. Server side processing will be faster, eases the processing burden on the Macromedia Flash Player (and the viewer’s CPU), and will become increasingly important as the audience for low-memory portable devices, such as PDA’s and cell phones, increases.

Avoid lengthy scripts that may prevent the user from interacting with the content. If it’s necessary to process a lot of information, consider breaking the processing into chunks and distributing the task across a series of frames. One frame can do the initial calculation, set up the necessary variables, then finish processing and rendering the visuals. In the next frame pick up those variables, perform more calculation, then again allow the frame to process and render. This type of “distributed processing” may be more difficult to write and debug, but will insure a more consistent experience for your users.

On a similar note, there may be simple scripts that are being evaluated on every frame. Consider if it might be possible to run these scripts once, save their results and call them when needed instead of rerunning the script every time.

If you’re controlling a large number of movie clip instances using ActionScript, consider how many times you might be repeating the same routines or resetting the same variables. Can that information be broken out into a single custom Function? Is it possible to use a function to calculate that information once and pass it back to the clip instances instead of processing the function multiple times?

For Function details, see the definition of Function in the online Flash Dictionary.

In some cases, it may be possible to achieve faster script processing by using deprecated Macromedia Flash 4 ActionScript syntax. This might achieve an immediate goal of faster code processing, but keep in mind that deprecated syntax is deprecated for a reason. There is no guarantee that the underlying mechanisms used to achieve that speed will be around in the next revision of the Flash Player, and an extensive recoding might be necessary.

Note: Macromedia Flash Player 6 includes significantly improved ActionScript performance, eliminating virtually all the circumstances in which it would be necessary to use Macromedia Flash 4 ActionScript syntax.
**Test Movie, Welcome Screen don't function correctly | CS5 with FileVault enabled**

**Issue**
When you use FileVault in Mac OS X 10.3 and later, the Welcome Screen and any WindowSWF panels (Project and Code Snippet panels) are blank. You also receive the following error when using Test Movie:

```
Error opening URL 'file:///Volumes/username/Library/Caches/TemporaryItems/Untitled%2D1.swf
<file:///\Volumes\username\Library\Caches\TemporaryItems\Untitled%2D1.swf> ' 
```

**Solution**
Turn off FileVault by choosing System Preferences > Security > FileVault tab > Turn Off FileVault.

If you can’t disable FileVault, Adobe provides an **AuthPlayLib.bundle** to resolve this issue. Follow the steps below:

1. Quit Flash.
2. Delete AuthPlayLib.bundle from /Applications/Adobe Flash CS5/Common/Configuration.
3. Download AuthPlayLib.zip below to your computer and unzip it.
4. Copy the unzipped AuthPlayLib.bundle into /Applications/Adobe Flash CS5/Common/Configuration.
5. Restart Flash.

---

| Bitmaps are fixed-resolution images. They look best when one pixel of source imagery exactly matches one pixel of screen display. Because SWF streams and scales, it’s rare to have this type of one-to-one correspondence. This means you’re asking the computer to create new screen pixels from groups of source pixels and draw them onscreen every time the playback moves to a new frame. Bitmap-heavy projects can be slower to redraw because of this. Make use of vectors and bitmap fills whenever possible to avoid this. Also, it may seem simple to mention it, but any item used more than once in the SWF should be inside a symbol. This includes bitmaps, imported vector art and art created in Flash. |
| Looping: tight loops can be problematic. Give the player a chance to break out of the loop so it can process other actions. |
| Minimize simultaneous streaming whenever possible. Multiple streams cost processing power and bandwidth. If multiple streams of audio are necessary or multiple SWF’s must be loaded, use a preloading technique at the beginning of the movie. This costs more time up front, but at that point it’s possible to let the viewer know that something is happening. Later in the movie a lag might be interpreted as a system problem or may lose the viewer’s attention. |
Supported file formats | Flash Professional CS5.5

The following tables list the file formats that Flash Professional CS5.5 can open, import, export, publish, or save. For more information about specific support for each file format, see Flash Professional CS5.5 Help.

**File formats Flash Professional CS5.5 can open**

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe Flash authoring File</td>
<td>FLA</td>
</tr>
<tr>
<td>ActionScript File</td>
<td>AS</td>
</tr>
<tr>
<td>Flash XML File</td>
<td>XML</td>
</tr>
<tr>
<td>Compiled Flash file</td>
<td>SWF</td>
</tr>
<tr>
<td>ActionScript Communication File</td>
<td>ASC</td>
</tr>
<tr>
<td>Flash JavaScript File</td>
<td>JSFL</td>
</tr>
</tbody>
</table>

**Outdated file formats**

Flash Professional CS5.5 can no longer import FreeHand, PICT, PNTG, SGI, and TGA files. Flash no longer exports EMF files, WMF files, WFM image sequences, BMP sequences, or TGA sequences.

**Graphic file formats Flash Professional CS5.5 can import**

<table>
<thead>
<tr>
<th>File Format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe Illustrator (version 10 or earlier)</td>
<td>AI</td>
</tr>
<tr>
<td>Adobe Photoshop</td>
<td>PSD</td>
</tr>
<tr>
<td>AutoCAD 10 DXF</td>
<td>DXF</td>
</tr>
<tr>
<td>Bitmap</td>
<td>BMP, DIB</td>
</tr>
<tr>
<td>Enhanced Windows Metafile (Windows only)</td>
<td>EMF</td>
</tr>
<tr>
<td>FutureSplash Player</td>
<td>SPL</td>
</tr>
<tr>
<td>GIF and animated GIF</td>
<td>GIF</td>
</tr>
<tr>
<td>JPEG</td>
<td>JPG, JPEG</td>
</tr>
<tr>
<td>PNG, including Fireworks PNG files</td>
<td>PNG</td>
</tr>
<tr>
<td>Flash Player 6/7</td>
<td>SWF</td>
</tr>
<tr>
<td>Windows Metafile</td>
<td>WMF</td>
</tr>
<tr>
<td>Adobe XML Graphic File</td>
<td>FXG</td>
</tr>
</tbody>
</table>

You can import the following bitmap file formats into Flash Professional CS5.5 on Macintosh and Windows if QuickTime 4 or later is installed:

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuickTime Image</td>
<td>QTIF</td>
</tr>
<tr>
<td>TIFF</td>
<td>TIF, TIFF</td>
</tr>
</tbody>
</table>
## Audio file formats Flash Professional CS5.5 can import

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe Soundbooth</td>
<td>ASND</td>
</tr>
<tr>
<td>Wave (Windows only)</td>
<td>WAV</td>
</tr>
<tr>
<td>Audio Interchange File Format (Mac OS only)</td>
<td>AIFF</td>
</tr>
<tr>
<td>MP3</td>
<td>MP3</td>
</tr>
</tbody>
</table>

You can import the following audio file formats into Flash Pro CS5.5 on Macintosh and Windows if QuickTime 4 or later is installed:

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Interchange File Format</td>
<td>AIF, AIFF</td>
</tr>
<tr>
<td>Sound Designer II (Macintosh only)</td>
<td>SD2</td>
</tr>
<tr>
<td>Sound Only QuickTime Movies</td>
<td>MOV, QT</td>
</tr>
<tr>
<td>Sun AU</td>
<td>AU</td>
</tr>
<tr>
<td>System 7 Sounds (Macintosh only)</td>
<td>SND</td>
</tr>
<tr>
<td>Wave</td>
<td>WAV</td>
</tr>
</tbody>
</table>

## Video file formats Flash Professional CS5.5 can import

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuickTime Movie</td>
<td>MOV, QT</td>
</tr>
<tr>
<td>Video for Windows</td>
<td>AVI</td>
</tr>
<tr>
<td>MPEG</td>
<td>MPG, M1V, M2P, M2T, M2TS, MTS, TOD, MPE, MPEG</td>
</tr>
<tr>
<td>Digital Video</td>
<td>DV, DVI</td>
</tr>
<tr>
<td>Video for Adobe Flash</td>
<td>FLV, F4V</td>
</tr>
<tr>
<td>3GPP/3GPP2 for Mobile Devices</td>
<td>3GP, 3GPP, 3GP2, 3GPP2, 3P2</td>
</tr>
<tr>
<td>MPEG-4</td>
<td>MP4, M4V, AVC</td>
</tr>
</tbody>
</table>

## File formats Flash Professional CS5.5 can export

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compiled Flash Movie</td>
<td>SWF</td>
</tr>
<tr>
<td>PICT and PICT sequence (Macintosh only)</td>
<td>PCT, PIC</td>
</tr>
<tr>
<td>Enhanced Windows Metafile and EMF sequence (Windows only)</td>
<td>EMF</td>
</tr>
<tr>
<td>Windows Metafile and WMF sequence (Windows only)</td>
<td>WMF</td>
</tr>
<tr>
<td>Bitmap and BMP sequence (Windows only)</td>
<td>BMP, DIB</td>
</tr>
<tr>
<td>GIF, GIF sequence, and animated GIF</td>
<td>GIF</td>
</tr>
<tr>
<td>JPEG and JPEG sequence</td>
<td>JPG, JPEG</td>
</tr>
</tbody>
</table>
File formats Flash Professional CS5.5 can publish

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compiled Flash and Flash Lite File</td>
<td>SWF</td>
</tr>
<tr>
<td>Compiled AIR File</td>
<td>AIR</td>
</tr>
<tr>
<td>HTML</td>
<td>HTML</td>
</tr>
<tr>
<td>GIF</td>
<td>GIF</td>
</tr>
<tr>
<td>JPEG</td>
<td>JPG, JPEG</td>
</tr>
<tr>
<td>PNG</td>
<td>PNG</td>
</tr>
<tr>
<td>Windows Projector</td>
<td>EXE</td>
</tr>
<tr>
<td>Macintosh Projector</td>
<td>APP</td>
</tr>
<tr>
<td>Compiled Flash Library/Script</td>
<td>SWC</td>
</tr>
<tr>
<td>Compiled Flash Player Flex-signed RSL File</td>
<td>SWZ</td>
</tr>
</tbody>
</table>

File formats Flash Professional CS5.5 can save

<table>
<thead>
<tr>
<th>File format</th>
<th>Filename extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe Flash CS5 File</td>
<td>FLA</td>
</tr>
<tr>
<td>Adobe Flash CS4 File</td>
<td>FLA</td>
</tr>
<tr>
<td>Adobe Flash CS5 Uncompressed Document</td>
<td>XFL</td>
</tr>
<tr>
<td>ActionScript File</td>
<td>AS</td>
</tr>
<tr>
<td>XML File</td>
<td>XML</td>
</tr>
<tr>
<td>ActionScript Communication File</td>
<td>ASC</td>
</tr>
<tr>
<td>Flash JavaScript File</td>
<td>JSFL</td>
</tr>
</tbody>
</table>

Slow opening transition when editing a symbol in Flash on Windows Vista
Issue
When double-clicking or editing a symbol currently located on the stage in an Adobe Flash document, you will see a slow animation of an expanding dashed box outline before the symbol is opened.

Reason
This is a known issue and is related to an incompatibility with Flash and Windows Vista's Aero graphics style.

Solution
To prevent this behavior you will need to either:
• disable Aero or
• disable desktop composition specifically for Flash
To disable desktop composition (Aero) specifically for Flash:
1 Right-click on the Flash icon (or a shortcut to Flash) and select properties.

2 In the resulting Flash Properties dialog, select the Compatibility tab.
3 In the Settings portion of the Compatibility tab, check the Disable desktop composition option.

4 Restart Flash if currently open.
Now, upon launching Flash, Windows will exit Aero. Without Aero enabled, editing symbols will no longer exhibit this behavior.

Set variables in a Flash movie
A variable is a container that holds information, such as numerical or string data. This TechNote outlines the five main ways to set variables in a Flash movie. For more information on variables, refer to page 191 of the Flash 4 manual, "Setting and identifying variables".

There are five main ways to set variables in Flash:
• Using the action Set Variable
• Using an editable text field
• Using the action Load Variables
• Appended on a query string in HTML tags
• Using JavaScript methods

Set Variable
Use the Set Variable action in Flash. This example uses a frame action, though it can also be done using a button action.

To set a variable using "Set Variable":
1 Open a new document in Flash.
2 Highlight Frame 1 in the timeline.
3 Choose Modify > Frame from the menu.
4 In the Actions tab, click the "+" sign, and choose "Set Variable".
5 In the "Variable" field, type "text" without quotes. This is the variable name.
6 In the "Value" field, type "hello" without quotes.
7 Click OK.
8 Choose Control > Test Movie to test this. The variable named "text" will be assigned the specified value "hello" when the movie plays the first frame of the timeline. Also, in this example, setting the variable will not show any visual change in the movie. However, variables and their values can be previewed in Test Movie mode by choosing Control > List Variables.

Note: To the right of the "Variable" and "Value fields", there is a pop-up menu, which contains "String Literal", "Expression", and "Expression Editor". The first time a variable is set, make sure the variable field is set to "String Literal". For more information creating expressions, please refer to page 196 of the Flash 4 manual.

Use a Text Field

Text fields in Flash can be used to assign a value to a variable by user text entry, or to display the value of a variable. For more information on creating text fields, see page 117 of the manual for more information on text fields, and Using Type Blocks and Text Fields in Flash (TechNote 14154).

To set a variable using a text field:
1 Open a new document in Flash.
2 Select the text tool.
3 Click the Text Field Modifier. (See image below)
4 Using this tool, click and drag to create a box to the desired width and height of the text field.
5 Choose Modify > Text Field from the menu, to access the Text Field Properties dialog box.
6 In the field marked "Variable", type "text" (without quotes). Click OK.
7 Choose Control > Test Movie. Any value entered into the text field will become the value of "text".

Note: A value you assign to a variable will also display in a text field with the same variable value. (For instance, if "text" is set to "hello", the word "hello" will display in the text field with variable name "text".)

Load Variables

Variables can be obtained from a remote file by using the action Load Variables. The remote file can be a text file or a server-side application such as ASP, CGI or ColdFusion. This example uses a text file as the data source, with the command issued by a frame action.

To obtain external data using Load Variables:
1 Create a text file (using SimpleText or Notepad) containing only the text "text=hello" without quotes. Save this text file as "text.txt" to your working folder.
2 Open a new document in Flash.
3 Highlight Frame 1 in the timeline.
4 Choose Modify > Frame from the menu.
In the Actions tab, click the "+" sign, and choose "Load/Unload Movie" from the pop-up menu.
On the right, choose the "Load Variables into Location" radio button.
In the "URL" field, type the name of the text file "text.txt".
In the "Level" field, type the number 0. Click OK.
Create a text field on the stage with variable name "text". This will demonstrate when the data has been loaded.
Publish the movie to the same folder as the text file "text.txt".
When the movie plays Frame 1, the movie will pull the variable and value pair "text=hello" from your data source into the current timeline. Additional data can be added to this by using the syntax specified on page 183.

Note: Special characters (such as punctuation, numeric operators, and other non-alphanumeric data) must be URL encoded to be translated into the Flash movie. For more information, see URL Encoding: Reading special characters from a text file (TechNote 14143).

On a Query String in HTML tags
Variables can be passed from an HTML page into the inserted Flash movie that it encompasses. This approach requires that tags are written into the HTML source that address both Internet Explorer and Netscape. The variable information is passed to the main timeline of the Flash movie as soon as the SWF loads. The example below uses a text field on the main timeline to display the variable that is being passed.

To pass variables on a query string to a Flash movie in HTML:
1 Create a new Flash movie and save the file with the name "movie.fla". Create a text field with variable name "text".  
(See above for steps)
2 Publish both Flash (movie.swf) and HTML (movie.html) files.
3 Open the movie.html file with a text editor such as SimpleText or Notepad.
4 Find the OBJECT tag. Look for this tag:<PARAM NAME=movie VALUE="movie.swf">
5 At the end of "movie.swf", add "?text=hello" It should look like this:<PARAM NAME=movie VALUE="movie.swf?text=hello">
6 Find the EMBED tag. Look for this:<EMBED href="/support/flash/ts/documents/movie.swf"
7 Again, replace the filename "movie.swf" with "movie.swf?text=hello". It should look like this:<EMBED href="/support/flash/ts/documents/movie.swf?text=hello"
8 Save the file as "movie.html", replacing the old file.
9 When you open the HTML file in a browser, the text field "text" in the Flash movie should display the value "hello".

Note: Multiple variables can be passed with this syntax: "movie.swf?variable1=value1&variable2=value2". The value must be assigned in both the OBJECT and EMBED tags for this method to work in all browsers. Also, Internet Explorer on the Macintosh will not display the movie locally. To preview this sample in that browser, upload the SWF and HTML files to a server and type in the full "http://" address.

Using JavaScript
It is possible to pass variables to a Flash movie from JavaScript in the HTML page. This is just one example of a strategy that can be used in many ways. In the example below, a link in the HTML page passes a variable when clicked to a text field in the Flash movie. This method will not work on Internet Explorer for the Macintosh.

To create this sample, do the following:
1 Open a new Flash document.
2 Create a text field with variable name "text". (See above for steps)

3 Add an additional frame to the Flash movie so that it has at least two frames.

4 Save the document as "movie.fla" and publish your document (using the "Default" HTML template) to create the Flash movie ("movie.swf") and HTML page ("movie.html").

5 Open the "movie.html" file with a text editor such as SimpleText or Notepad.

6 Between the HEAD tags, insert this code. Be sure to copy it exactly:
   ```html
   <script language="JavaScript"><!-- function PassFlash(){
   window.document.movie.SetVariable("text", "hello"); } //-->
   
   </script>
   ```

7 In the EMBED tag, look for the HEIGHT and WIDTH parameters. Insert the following parameter (if it is not already in there):`NAME=movie`

8 After this has been added, insert the follow code directly following this: `swLiveConnect=true`

9 Just above the closing body tag ("</body>"), insert this code: `<a href="#" onClick="PassFlash()">Pass The Variable</a>`

10 Save the document as "movie.html", replacing the old version of this file.

11 Open "movie.html" in a browser. To pass the variable to the movie, click the text "Pass the Variable". The value of "text" becomes "hello", and the text field will display this new value.

Note: A complete description of JavaScript is beyond the scope of this TechNote. Complete technical documentation is available from numerous third-party JavaScript books and online resources such as Webmonkey's Programming JavaScript. If you use a JavaScript method to communicate with a Flash movie, additional logic may be necessary to make sure issues such as browser type, layers and such are considered when implementing a script.

Additional Information

Using variables in Flash can create powerful and scalable movies that can change to reflect user choices, updates and events. Writing expressions adds new levels of complexity to authoring in Flash. For more instructions on many Flash scripting and design techniques, please visit the Flash Support Center and view related TechNotes. Enter keyword topics into the search field to access a wide variety of articles on beginning and advanced topics.

Re-create preferences

Re-creating the Flash Professional preferences file sometimes resolves unexpected issues or unusual performance, such as a system freeze.

Re-create the preferences file to restore the following items to the default settings:

- All preferences
- All workspace settings
- All panel positions

Last updated 11/6/2015
• All behaviors
• All Project settings
• All cached custom classes saved to the Flash programs folder 'First Run' folder
• All custom commands such as Motion XML in the commands menu

**Save custom settings (optional)**
To save custom commands and other settings, create a copy of this folder:

**CS6**
- Windows: C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS6/<language>/
- Windows Vista & Windows 7: C:/Users/<user name>/AppData/Local/Adobe/Flash CS6/<language>/
- Mac OS X: /User/<user name>/Library/Application Support/Adobe/Flash CS6/<language>/

**CS5.5**
- Windows: C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS5.5/<language>/
- Windows Vista & Windows 7: C:/Users/<user name>/AppData/Local/Adobe/Flash CS5.5/<language>/
- Mac OS X: /User/<user name>/Library/Application Support/Adobe/Flash CS5.5/<language>/

**CS5**
- Windows: C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS5/<language>/
- Windows Vista & Windows 7: C:/Users/<user name>/AppData/Local/Adobe/Flash CS5/<language>/
- Mac OS X: /User/<user name>/Library/Application Support/Adobe/Flash CS5/<language>/

**CS4**
- Windows: C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS4/<language>/
- Windows Vista & Windows 7: C:/Users/<user name>/AppData/Local/Adobe/Flash CS4/<language>/
- Mac OS X: /User/<user name>/Library/Application Support/Adobe/Flash CS4/<language>/

To restore custom settings, copy the commands folder (and Projects or other needed custom settings) back into the original location and restart Flash Professional.

**Re-create the preferences file**
**Note:** You can skip the following steps if you do not want to save your old preferences. You can start the application while holding Ctrl+Alt+Shift to delete and reset the preferences to the default.

To delete the preferences on Windows, delete the following registry entries.
Warning: The following solution involves the Windows System Registry. Editing or manipulating the registry incorrectly can result in serious system damage which sometimes requires reinstallation of the operating system. If you are not comfortable editing the registry, then take your system to a professional. If you choose to proceed, make sure that you create a complete system backup and a Windows System Restore Point before proceeding. Adobe Systems cannot be held responsible for damage resulting from this information.

CS4

2. Type regedit and click OK.
3. In Registry Editor, right-click the My Computer/HKEY_CURRENT_USER/Software/Adobe/Flash10 registry key and delete it.

CS6, CS5.5, and CS5

2. Type regedit and click OK.
3. Do one of the following:
   • CS6: In Registry Editor, right-click the My Computer/HKEY_CURRENT_USER/Software/Adobe/Flash12 registry key and delete it.
   • CS5.5: In Registry Editor, right-click the My Computer/HKEY_CURRENT_USER/Software/Adobe/Flash11.5 registry key and delete it.
   • CS5: In Registry Editor, right-click the My Computer/HKEY_CURRENT_USER/Software/Adobe/Flash11 registry key and delete it.

When you delete the preferences files, Flash creates new files with default settings the next time you start Flash. To re-create the preferences files, delete the following items:

CS6

Windows XP

• C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS6/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Windows Vista and Windows 7

• C:/Users/<username>/AppData/Local/Adobe/Flash CS6/
• C:/Windows/system32/Macromed\Flash\FlashAuthor.cfg

Mac OS

• /Users/<username>/Library/Preferences/Flash CS6 Preferences
• /Users/<username>/Library/Preferences/Flash 12 MRU
• /Users/<username>/Library/Preferences/com.adobe.flsh-12.plist
• /User/<username>/Library/Application Support/Adobe/Flash CS6/
• /Library/Application Support/Macromedia/FlashAuthor.cfg

CS5.5

Windows XP

• C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS5.5/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Windows Vista and Windows 7
• C:/Users/<user name>/AppData/Local/Adobe/Flash CS5.5/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Mac OS
• /Users/<user name>/Library/Preferences/Flash CS5.5 Preferences
• /Users/<user name>/Library/Preferences/Flash 11.5 MRU
• /Users/<user name>/Library/Preferences/com.adobe.flsh-11.5.plist
• /User/<user name>/Library/Application Support/Adobe/Flash CS5.5/
• /Library/Application Support/Macromedia/FlashAuthor.cfg

CS5
Windows XP
• C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS5/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Windows Vista and Windows 7
• C:/Users/<user name>/AppData/Local/Adobe/Flash CS5/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Mac OS
• /Users/<user name>/Library/Preferences/Flash CS5 Preferences
• /Users/<user name>/Library/Preferences/Flash 11 MRU
• /Users/<user name>/Library/Preferences/com.adobe.flsh-11.0.plist
• /User/<user name>/Library/Application Support/Adobe/Flash CS5/
• /Library/Application Support/Macromedia/FlashAuthor.cfg

CS4
Windows XP
• C:/Documents and Settings/<user name>/Local Settings/Application Data/Adobe/Flash CS4/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Windows Vista & Windows 7
• C:/Users/<user name>/AppData/Local/Adobe/Flash CS4/
• C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Mac OS
• /Users/<user name>/Library/Preferences/Flash CS4 Preferences
• /Users/<user name>/Library/Preferences/Flash 10 MRU
• /Users/<user name>/Library/Preferences/com.adobe.flsh-10.0.plist
• /User/<user name>/Library/Application Support/Adobe/Flash CS4/
• /Library/Application Support/Macromedia/FlashAuthor.cfg

**Note:** Do not remove the primary application from the following location:

• Windows: C:/Program Files/Adobe Flash CS4/
• Mac OS: /Applications/Adobe Flash CS4/
• Start Flash CS4/CS5/CS5.5 Professional. The preferences file is re-created during startup.

### Problems using Adobe Flash authoring across local area networks

**Issue**

When using Adobe Flash across a local area network (LAN) and networked drives/folders, you may experience any of the following problems:

• Flash crashes while performing a test movie on FLA files located on a networked drive or folder.
• FLA files get corrupted when opening from or saving to networked drives or folder.
• Flash does not reflect changes in custom class after compiling.
• Flash, Flash Video Encoder, or Adobe Media Encoder crashes or corrupts Flash Video (FLV) files while encoding source located on networked drives or folder.
• Flash Video Encoder or Adobe Media Encoder crashes or corrupts FLV files where the output folder is a networked drive or folder.
• Published Flash Player (SWF) files and projectors are unable to load content located on networked drives or folder.
• More than one instance of a SWF or Projector on client machines cannot play back FLV files located on a networked drive or folder.

**Reason**

The Adobe Flash IDE, FLV Encoder, Adobe Media Encoder and Flash Player were not designed to function across LANs.

**Solution**

Use of Flash files across local networks is not supported in any context. Published content should access data through a web server. All file sources should be opened and saved on the local system. Using Flash in such a scenario for project collaboration or content deployment is highly discouraged and may corrupt your source files.

If you need to work in a collaborative environment or store source files on a server, use the project panel and/or a third-party version control system.
Pause SWF file | Specific number of seconds

This TechNote provides an ActionScript 2.0 method for looping over a range of frames for a given amount of time. It can be tempting to use a while or for loop to achieve this end, but this method is generally a bad programming practice. A script executes until finished. If you stay within one script for multiple seconds, then the computer can't perform other tasks such as refresh the display or check for mouse events. In some cases, it can appear as if the computer crashed.

To stay in a range of frames for a given number of seconds, consider using something like the following script. Instead of staying stuck in one loop, it repeats over a series of frames, checking for whether a condition is fulfilled before proceeding.

Note: The code provided below is ActionScript 2.0. ActionScript 2.0 and 3.0 are incompatible with each other.

```actionscript
// Set the following two values for how long you want to // loop, and how many frames to loop: pauseDuration = 5 * 1000; framesInLoop = 10;
if (startTime == null)
{
    startTime = getTimer();
gotoAndPlay(_currentframe - framesInLoop);
}
else
{
    lapsedTime = getTimer() - startTime;
    if (lapsedTime < pauseDuration)
    {
        gotoAndPlay(_currentframe - framesInLoop);
    }
    else
    {
        startTime = null;
    }
}
```

When the script runs, it first checks whether the variable named startTime has been initialized. If not, it sets it to the current time. Each time the playhead reaches the frame containing the script, it checks whether the elapsed time has exceeded the desired pause duration. It loops back if the time has not yet been reached.

To use this script, enter (or copy and paste) the code example above into the last frame of the range of frames included in the loop. Then, set the variables for how long the loop continues (pauseDuration) and how many frames are included in the loop (framesInLoop). These variables appear at the top of the script.

For example, the sample script above causes the playhead to loop over ten frames for 5 seconds (5 * 1000 milliseconds). Placing this script on a keyframe at frame 20 causes the playhead to loop over frames 10-20 for 5 seconds before continuing.

In summary, a script executes until done—nothing else can happen while it's stuck in a while or for loop. For perceptible delays, try regularly testing for a condition in an open frame loop, instead of staying stuck in a tight scripting loop.

Note: In a multi-scene movie, this code works only in the first scene. The reason is that the _currentFrame property is not scene-specific. For example, if a movie contains three scenes each containing ten frames, _currentFrame evaluates to "30" on the last frame of scene three, rather than "10." The gotoAndPlay() function is scene-specific, however.
How to "park" a movie clip so it can be preloaded and reused

"Parking" a movie clip is a useful technique. A parked movie clip can sit invisibly on stage waiting for instructions. This allows the movie clip to be preloaded along with the main timeline and avoids playback delays later in the movie.

Follow the steps below to create the movie clip:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create a new movie clip using Insert &gt; New Symbol. Choose Movie Clip as the symbol's behavior.</td>
</tr>
<tr>
<td>2</td>
<td>Double click the keyframe in frame 1 and add a Stop action to the frame. Do not place any graphics in this frame.</td>
</tr>
<tr>
<td>3</td>
<td>Add a new keyframe at frame 2 (or at any frame after frame 1). The actual animation starts in this keyframe and can contain graphics or other movie clips. For convenience, add a frame label to this keyframe such as &quot;Play&quot;.</td>
</tr>
<tr>
<td>4</td>
<td>Choose Edit &gt; Edit Movie to return to edit movie mode.</td>
</tr>
<tr>
<td>5</td>
<td>Open the Library and drag an instance of this movie clip onto the stage. Because there are no graphics in the first frame of the movie clip it will be invisible in edit movie mode. Also, because the clip has a Stop action on its first frame nothing will display by default when previewing the movie.</td>
</tr>
<tr>
<td>6</td>
<td>Using the Instance panel, give the movie clip an instance name.</td>
</tr>
</tbody>
</table>

To make use of this parked movie clip you use the tellTarget or with actions to send the movie clip to the frame that contains the actual graphics or animation. For example, to "activate" the movie clip from a button instance, attach the following script to the instance:

```actionscript
on (release) { tellTarget ("instanceName") { gotoAndPlay ("Play"); } }
```

Or, using the withaction,

```actionscript
on (release) { with (instanceName) { gotoAndPlay ("Play"); } }
```

Similarly, to "deactivate" the parked clip, or send it back to the empty keyframe in frame 1, use a similar action to send the clip to frame 1, which contains no content.
Additional Information For more information on using the tellTarget action see How to use the Tell Target action (TechNote 13479). For more advanced information on working with movie clips, please refer to the Flash MX Application Development Center at Macromedia’s Designer and Developer Center.

Note: Although a similar effect to the one described here can be achieved by changing the _visible property of a movie clip instance, this occasionally causes the movie clip to “flash” briefly before becoming invisible. Parking a movie clip can provide more consistent display performance.

OBJECT tag syntax | Flash Professional

Introduction
Correctly displaying a SWF file in a browser requires an HTML file. The browser loads the HTML file, and then loads the SWF file that is specified in the HTML code. It’s necessary that the HTML page contains an OBJECT tag that specifies the location of the actual Adobe Flash SWF file.

The HTML OBJECT tag directs the browser to load Adobe Flash Player and then use it to play your SWF file.

The Publish command in Flash Professional automatically creates an HTML file that contains the required tags for web browsers to play the published SWF file. There are several different HTML publishing templates you can choose from in the Publish dialog box (File > Publish). See Flash Help for more information about Publish settings (Flash CS5) or Publish settings (Flash CS5.5).

Add the OBJECT tag manually
The HTML files created by Flash Professional contain only the code required to display your SWF file. To add your SWF file to an HTML page that contains other elements such as text or graphics, paste the following code into the HTML document. Paste this code somewhere inside the BODY tag of the HTML file.

To add the OBJECT tag manually, do the following:

1. Copy the HTML code below and paste it into your HTML. Place the code in the HTML file in the location where you want the SWF file to appear in the web page.
2 Edit the attributes of the OBJECT tag for your SWF file.

- Change the HEIGHT and WIDTH parameters to match the height and width of the Stage. To automatically scale the SWF file to consume a specific percentage of the browser window, enter percentage values for height and width instead.
- Change all instances of "movie_name.swf" where it appears in the OBJECT tag above to the actual path or URL of your SWF file.

The SWF file path can be relative to the location of your HTML file or the fully qualified, absolute path of your SWF file. For example:

- If the SWF file is in the same folder as the HTML file, you only need the filename of the SWF file.
- If the SWF file is in a subfolder, the path is something like "subfolder_name/movie_name.swf" (without the quotes).
- If the SWF file isn’t in the folder with the HTML file, use "./" for each level up from the folder with the HTML file. (Don’t include the quotes in the syntax.) So "./../separate_folder/movie_name.swf" goes up three levels from the folder with the HTML file before going down into the "separate_folder" to find the SWF file.
- A fully qualified path is the same thing as a complete URL (that is, http://www.myDomain.com/flash_content/movie_name.swf.)

You can also use the Publish command in Flash to create a basic HTML file containing the OBJECT tag. Then copy and paste the HTML code within the OBJECT tag into your own HTML file. The, the name of your SWF file is already correctly specified in the HTML code.

The code above is the minimum required to insert a SWF file in your HTML file. There are additional attributes you can specify for these tags to control the details of how Flash Player plays your SWF file. For information, see Flash OBJECT and EMBED tag attributes.

Test your HTML page in all web browsers you want your intended audience to be able to use.

**Note:** Dreamweaver, Adobe’s HTML authoring tool, can easily embed a SWF file into an HTML document for you. Dreamweaver allows full control of the design of the page that contains the SWF file. For more information, see the Dreamweaver product page.
Multiple instances of movie clip with armature trace only one armature

Issue
When multiple instances of a movie clip with an armature are placed on stage, ActionScript reports only one armature.

For example:

1. Create a movie clip with an armature.
2. Drag three instances of that clip to the stage.
3. Add a frame script: import fl.ik.*; trace(IKManager.numArmatures);

When you test it using Test Movie, that trace returns “1” no matter how many instances of that movie clip are on stage.

Solution

Use IKArmature.registerElements to gain control of the armature inside each movie clip instance

To see an example, download kb405649.zip, which uses the following timeline script:

```actionscript
import fl.ik.*;

// there is only 1 armature, although there are 2 instances of the containing Movieclip on stage

trace(IKManager.numArmatures);

// Store a reference to the armature

var arm:IKArmature = IKManager.getArmatureAt(0);

// Get reference to the last joint in the armature.

// We will move this joint.
```
var lastJoint:IKJoint = arm.rootJoint.getChildAt(0).getChildAt(0).getChildAt(0);

// Create the mover for the last joint.
var mv:IKMover = new IKMover(lastJoint, lastJoint.position);

var timer:Timer = new Timer(100);
timer.addEventListener(TimerEvent.TIMER, moveArmatures);

var pt:Point;

// You can move the armature within each Movieclip, but
// you need to alternate between them when registering them
// to a DisplayObject
function moveArmatures(evt:TimerEvent):void
{
    // move first armature
    arm.registerElements(mc1);
    pt = new Point(lastJoint.position.x, lastJoint.position.y - 5);
    mv.moveTo(pt);

    // move second armature
    arm.registerElements(mc2);
    pt = new Point(lastJoint.position.x, lastJoint.position.y - 5);
    mv.moveTo(pt);
}

timer.start();
Additional information
Armatures are designed to work this way. Technically, no matter how many instances of that symbol are on stage, there is still only one armature.

If you target that armature with ActionScript, the code is expected to affect that single armature in every instance on stage.

The result is that there is no intuitive way to target each instance’s armature with ActionScript.

MovieClip coordinates off on first frame of 3D object | QuickTime (.mov) file | Flash Professional CS5.5

Issue
When you export a 3D object from Flash CS5.5 to a QuickTime (.mov) file, the clip’s coordinates are off on the first frame of the timeline. The frame appears in the upper-right corner of the stage, rather than in the center as expected.

Solution
Insert an empty frame at the start of the movie, or an empty scene at the beginning of the FLA file.

Manually remove Flash Professional CS5.5, CS5, CS4

Use the uninstallers
Uninstalling Adobe Flash Professional CS5.5, CS5 or CS4 is best done using the uninstallers. Before you uninstall, close all applications currently running on your system—including other Adobe applications, Microsoft Office applications, and browser windows. Then do one of the following:

• In Windows XP, open the Windows Control Panel and double-click Add or Remove Programs. Select the product that you want to uninstall, click Change/Remove, and then follow the onscreen instructions.

• In Windows Vista, open the Windows Control Panel and double-click Programs and Features. Select the product that you want to uninstall, click Uninstall/Change, and then follow the onscreen instructions.

• Note: Mac OS has new uninstall functionality. DO NOT drag applications to the trash to uninstall them. To safely uninstall on Mac OS X, double-click the product installer in Applications/Utilities/Adobe Installers. Authenticate as an administrator, then select Remove Components and follow the onscreen instructions.

Troubleshoot uninstall
For troubleshooting uninstallation issues, see Resolve installation problems with CS5 Cleaner Tool (cpsid_82947).
Manual uninstall information

Note: Adobe doesn’t recommend manual removal of Flash Professional CS5.5, CS5 or CS4. The following information is provided for IT professionals for troubleshooting purposes.

Flash Professional CS5.5, CS5 or CS4 installs files in the following locations:

**Primary application location for Adobe Flash CS4 Professional**
Windows: C:/Program Files/Adobe Flash CS4/  Mac OS: /Applications/Adobe Flash CS4/

**Primary application location for Adobe Flash Professional CS5**
Windows: C:/Program Files/Adobe Flash CS5/ Mac OS: /Applications/Adobe Flash CS5/

**Primary application location for Adobe Flash Professional CS5.5**
Windows: C:/Program Files/Adobe Flash CS5.5/ Mac OS: /Applications/Adobe Flash CS5.5/

**Startup scripts for Adobe Flash CS4 Professional**
Windows: C:/Program Files/Common Files/Adobe/Startup Scripts CS4/Flash/Flash-10.0.jsx Mac OS: /Library/Application Support/Adobe/Startup Scripts CS4/Flash/Flash-10.0.jsx

**Startup scripts for Adobe Flash Professional CS5**
Windows: C:/Program Files/Common Files/Adobe/Startup Scripts CS5/Flash/Flash-11.0.jsxMac OS: /Library/Application Support/Adobe/Startup Scripts CS5/Flash/Flash-11.0.jsx

**Startup scripts for Adobe Flash Professional CS5.5**
Windows: C:/Program Files/Common Files/Adobe/Startup Scripts CS5.5/Flash/Flash-11.5.jsxMac OS: /Library/Application Support/Adobe/Startup Scripts CS5.5/Flash/Flash-11.5.jsx

**Files installed to Mac OS Only for Adobe Flash CS4 Professional**
/Users/%username%/Library/Preferences/Flash CS4 Preferences /Users/%username%/Library/Preferences/Flash 10 MRU /Users/%username%/Library/Preferences/com.adobe.flash.plist /User/%username%/Library/Application Support/Adobe/Flash CS4/ /Library/Application Support/Macromedia/FlashAuthor.cfg

**Files installed to Mac OS Only for Adobe Flash Professional CS5**
/Users/%username%/Library/Preferences/Flash CS5 Preferences /Users/%username%/Library/Preferences/Flash 11 MRU /Users/%username%/Library/Preferences/com.adobe.flash.plist /User/%username%/Library/Application Support/Adobe/Flash CS5/ /Library/Application Support/Macromedia/FlashAuthor.cfg

**Files installed to Mac OS Only for Adobe Flash Professional CS5.5**
/Users/%username%/Library/Preferences/Flash CS5.5 Preferences /Users/%username%/Library/Preferences/Flash 11.5 MRU /Users/%username%/Library/Preferences/com.adobe.flash.plist /User/%username%/Library/Application Support/Adobe/Flash CS5.5/ /Library/Application Support/Macromedia/FlashAuthor.cfg
Files installed to Windows only for Adobe Flash CS4 Professional
C:/Documents and Settings/username/Local Settings/Application Data Adobe/Flash CS4/
C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Files installed to Windows only for Adobe Flash Professional CS5
C:/Documents and Settings/username/Local Settings/Application Data Adobe/Flash CS5/
C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Files installed to Windows only for Adobe Flash Professional CS5.5
C:/Documents and Settings/username/Local Settings/Application Data Adobe/Flash CS5.5/
C:/Windows/system32/Macromed/Flash/FlashAuthor.cfg

Files installed to both Mac OS and Windows for Adobe Flash CS4 Professional


[AdobeCommon]/Help/it_IT/AS2LCR/Flash_10.0/ [AdobeCommon]/Help/it_IT/AS3LCR/Flash_10.0/ [AdobeCommon]/Help/it_IT/Flash/10.0_ExtendingFlash/ [AdobeCommon]/Help/it_IT/Flash/10.0_UsingFlash/ [AdobeCommon]/Help/it_IT/Flash/10.0_Welcome/


[AdobeCommon]/Help/nl_NL/AS2LCR/Flash_10.0/ [AdobeCommon]/Help/nl_NL/AS3LCR/Flash_10.0/ [AdobeCommon]/Help/nl_NL/Flash/10.0_ExtendingFlash/ [AdobeCommon]/Help/nl_NL/Flash/10.0_UsingFlash/ [AdobeCommon]/Help/nl_NL/Flash/10.0_Welcome/


Files installed to both Mac OS and Windows for Adobe Flash Professional CS5


[AdobeCommon]/Help/it_IT/AS2LCR/Flash_11.0/ [AdobeCommon]/Help/it_IT/AS3LCR/Flash_11.0/ [AdobeCommon]/Help/it_IT/Flash/11.0_ExtendingFlash/ [AdobeCommon]/Help/it_IT/Flash/11.0_UsingFlash/ [AdobeCommon]/Help/it_IT/Flash/11.0_Welcome/ [AdobeCommon]/Help/it_IT/FlashLite/ [AdobeCommon]/Help/it_IT/FlashPlatform/
Files installed to both Mac OS and Windows for Adobe Flash Professional CS5.5


[AdobeCommon]/Help/it_IT/AS2LCR/Flash_11.5/ [AdobeCommon]/Help/it_IT/AS3LCR/Flash_11.5/ [AdobeCommon]/Help/it_IT/Flash/11.5_ExtendingFlash/ [AdobeCommon]/Help/it_IT/Flash/11.5_UsingFlash/ [AdobeCommon]/Help/it_IT/Flash/11.5_Welcome/[AdobeCommon]/Help/it_IT/Flash/CS5.5/[AdobeCommon]/Help/it_IT/FlashLite/ [AdobeCommon]/Help/it_IT/FlashPlatform/


[AdobeCommon]/Help/nl_NL/AS2LCR/Flash_11.5/ [AdobeCommon]/Help/nl_NL/AS3LCR/Flash_11.5/ [AdobeCommon]/Help/nl_NL/Flash/11.5_ExtendingFlash/ [AdobeCommon]/Help/nl_NL/Flash/11.5_UsingFlash/ [AdobeCommon]/Help/nl_NL/Flash/11.5_Welcome/[AdobeCommon]/Help/nl_NL/Flash/CS5.5/[AdobeCommon]/Help/nl_NL/FlashLite/ [AdobeCommon]/Help/nl_NL/FlashPlatform/


Last updated 11/6/2015
How to make a simple clock in Flash

The Date object introduced in Macromedia Flash 5 makes it simple to have a clock in a Flash movie. The following steps explain how to make a simple clock that displays the time in a Hours:Minutes:Seconds am/pm format. For steps on making a clock with separate hour, minute, and second hands see Using the Date object: Building a clock in Flash 5.

To make a Flash clock:

1. Create a new movie clip symbol (Insert > New Symbol).
2. Using the text tool, drag out a text field in this new symbol. This is the text field that will display the current time.
3. Select the text field with the Arrow tool, and open the Text Options panel (Window > Panels > Text Options).
4. In the Text Options panel, choose Dynamic Text from the pop-up menu, and give the text field a variable name oftime.
5 Double-click in the first frame of the symbol to open the Actions panel, and enter the following script:

```javascript
function getTime () {  var time = new Date();  var hour = time.getHours();  var minute =
    time.getMinutes();  var second = time.getSeconds();  var temp = ""+((hour>12) ? hour-12 :
    hour); temp += ((minute<10) ? ":0" : ":") + minute; temp += ((second<10) ? ":0" : ":") + second;
    temp += (hour>=12) ? " P.M." : " A.M.";  return temp; }
```

6 Place an instance of this new symbol on the main Timeline of the movie.

7 Select the new movie clip instance, and open the Actions panel (Window > Actions) if it’s not already open.

8 Enter the following script into the Actions panel:

```javascript
onClipEvent (enterFrame) {  time = getTime(); }
```

9 Test the movie (Control > Test Movie).

How it works

The getTime() function first creates a new Date object, which is used to get the current hour, minutes, and seconds.
Next, several ?: conditional operators (a compact form of an if..else statement) are used to determine the proper
formatting for the time string (HH:MM:SS PM/AM), and assign the string to the variable temp. For example, the
following line of code checks if the value of hours is greater than 12. If so, 12 is subtracted from hour and assigned to
temp. Otherwise, the value of hour is assigned to temp, as is.

```javascript
var temp = ""+((hour>12) ? hour-12 : hour);
```

For minute and seconds, similar logic is used to check if their values are less than 10. If so, a ":0" is appended to temp
before the minutes or seconds. Otherwise, the value of minute or second is appended to temp, as is. Finally, "P.M." is
appended to temp if the current hour is greater than or equal to 12, otherwise "A.M." is appended.

Note: If you don’t care to display seconds or an A.M./P.M. modifier in the time string, simply insert comments (//) at
the beginning of the function statements that append that information to temp.

Finally, in step 8, the statement within the onClipEvent(enterFrame) handler assigns the return value of getTime() to
time, the variable name assigned to the text field created in step 1. The enterFrame event is triggered as each frame of
the movie is played, so the time updates at each frame. This dependency on frame rate can effect the clock’s accuracy
over long periods, but should be fine for most applications, as long as the frame rate is 12 FPS or greater.

Additional information

For In versions of Flash Player 5 prior to 5,0,41,0, the getHours() function on Windows may return the incorrect hour.
See Incorrect hour returned by getHours() and getUTCHours() (TechNote 14964) for more information.

For another example of using movie clip events see Detecting a double-click in the Flash Player (TechNote 14461).

Leading ampersand breaks text file parsing in ActionScript 3.0

Issue

In ActionScript 3.0, you cannot use the URLVariables class to load a text file beginning with an ampersand.
Reason

In ActionScript 2.0, the LoadVars class can be used to load a text file beginning with this format: &name=peter
Technically the leading ampersand was an incorrect usage and should have been prevented the load. However, versions
of Flash Player prior to Flash Player 9 would silently ignore the ampersand.

ActionScript 3.0 is as standards-compliant as possible, and the leading ampersand may be interpreted as an incorrect
character when loaded by the URLVariables class. This may prevent the text file from parsing and may display the
following error message:

Error: Error #2101: The String passed to URLVariables.decode() must be a URL-encoded query string containing
name/value pairs.

Solution

To resolve the parsing problem for ActionScript 3.0, remove the leading ampersand from the text file being loaded. For
example, while &name=peter may fail, name=peter will parse correctly.

Additionally, the & character can be used to separate variables in a data file, for example, first=peter&last=strauss
However, any desired use of the & in the final destination variable must be URL-encoded using %26, for
example, var1=Bent&var2=Ben%26Jerrys

Large Flash applications compile incorrectly | Flash CS3, CS4, CS5, CS5.5

Issue

Large Adobe Flash CS3, CS4, CS5 and CS5.5 applications don’t compile. A blank Test Movie window appears or the
SWF file is blank.

Solution

Flash CS5 & CS5.5

The same property as in the CS4 solution below can be configured in the jvm.ini file in the location below:

<InstallLocation>/Common/First Run/ActionScript 3.0/jvm.ini

Open the jvm.ini file in your text editor of choice and find the following line:

-Xmx128m

Change the 128 to the suggested amount (256 or more depending on the application). Restart Flash Professional to give
the JVM the new starting amount of memory.

Note: A new warning dialog has been added to CS5.5 for users that encounter this issue. Below is what the warning
dialog will look like:
Clicking the OK button will open the jvm.ini file mentioned above in the Flash CS5.5 code editor and allow the user to change the -Xmx128m value. **Note:** You must quit and restart Flash Professional for the changes to take effect.

**Flash CS4**
Warning! By following the directions below, you modify the Windows System Registry. A mistake could cause serious system damage, which could require you to reinstall your operating system.

If you are not comfortable editing the registry, get assistance from your system administrator or another IT professional. If you choose to proceed, first back up your entire hard drive, and create a Windows System Restore Point.

An edit to the "JVM Max Heap Size" property in the ActionsInspector category can resolve this issue.

(Windows) Add a DWORD Value named "JVM Max Heap Size" with a value of "256" in the following location in the registry:

\HKEY_CURRENT_USER\Software\Adobe\Flash 10\ActionsInspector

(Mac OS) Add the following entry to the <ActionsInspector> element of the Flash CS4 Preferences file located in the /Users/<username>/Library/Preferences folder:

<JVM_Max_Heap_Size>256</JVM_Max_Heap_Size>

The number read in is interpreted as megabytes. The max heap size defaults to 128 megs. Adobe suggests an increase to 256 megs to resolve this issue. If this increase in memory does not resolve the issue, continue to increase by 256 megs.

**Note:** It is best practice to not increase this number by too much. If you increase it too much, Flash Professional can use up all available memory on the user’s computer. Adobe recommends that you only raise the heap size enough to fix the problem at hand.

**Recommendation for CS3 users**
When designing a Flash-based application, it is best practice to build it with multiple smaller objects which can be compiled independently.

1 Create custom components (SWC). Learn more about Creating ActionScript 3 Components.
2 Use various SWF files which can be accessed dynamically. Learn more on dynamically loading SWF files using the Loader Class (ActionScript 3)

These steps prevent the issue described in this document, and also speed up the compile time of your application.

**Additional information**
There is no unique number that qualifies as too much. This issue typically occurs in Flash-based applications made up of thousands of lines of code, numerous large objects, or a great number of small classes.

This issue occurs when the JVM compiler max heap size isn’t set high enough. Flash doesn’t have enough memory to complete certain compile operations. This issue can be solved in CS4 by editing a registry entry on Windows or by editing a preference file on a Macintosh. CS5 users can resolve this issue by editing the jvm.ini file as shown in the solution below.
Known issues | Flash Professional CS6

This document contains the current known issues for Adobe Flash Professional CS6. If you have questions that this document does not answer, see the Flash Help and Support page.

Known issues

Tools

• On Mac OS, when shortcuts for tools are started with ATOK IME on, Flash Professional CS6 tools don’t work on the Stage.

• Flash Professional CS6 crashes when converting more than two symbols into a symbol. This issue usually happens as a result of the computer being out of memory. To solve this issue, free up available memory. To free up memory, try restarting the system, or closing other open applications or documents. If neither of these options work, add more RAM to the system.

Code snippets

• You cannot type double-byte characters directly into the Code Snippets panel through IME.

XFL

• Flash CS5 files sometimes don’t open if they contain components in the Library that have been moved to a folder other than where originally created. (For example, the issue occurs when components are dragged from the Components panel into the Library panel.)

• Symbols saved in XFL format (compressed or uncompressed) are corrupted if its name has a special character. Don’t use special characters in symbol names.

• Windows only: You can’t save a FLA file in a Network/Desktop folder a second time after receiving an “unable to save” error message. To solve this issue, save the file as a FLA file instead.

AIR, AIR for Android, AIR for iOS

• An error occurs when you publish an AIR file if a surrogate pair Unicode character appears in the path to or filename of FLA file. The error also occurs when a surrogate pair Unicode character appears in the file or OS login name.

• When you use a Japanese Device font in a TLF vertical text field, Japanese symbols appear rotated on the Android runtime and AIR2.6.

• (Windows) When you publish an AIR for iOS application, an error occurs when the path to the FLA file includes a double-byte or high-ASCII character (not including the FLA filename).

• When you use a double-byte or high-ASCII character in the filename or path to any included file of an AIR for Android FLA file, the .apk file isn’t created.

• When you use the Japanese language environment of Mac OS, Flash crashes when a high-ASCII-European character is used in the AIR for Android (.apk) output filename.

Installation

• Flash crashes when installed at custom location with special characters.
Flash Builder
• Flash Builder always targets the highest version of Flash Professional installed.

Flash Professional Help
• Help does not launch from Flash Professional CS6 due to missing Help Configuration files. To access online Help resources, download and install Flash Pro CS6 Help Update.

Alternatively, you can install the Flash Professional CS6 Help Update using Adobe Application Manager.

1 In Flash Professional CS6, choose Help > Updates to launch the Adobe Application Manager.
2 Choose Adobe Flash Professional CS6 > Flash Pro CS6 Help Update.
3 Click Update.

Known issues with Flash Professional CS6 Update 12.0.2
• [Reference #3294338]: Error opening a FLA with ANE extension on a different computer from the one on which it is created.
• [Reference #3321319]: Remote debugging does not function with the default network interface for AIR 3.4.
• [Reference #3298386]: Error when using a FLA mounted on volume named as users.
• [Reference #3309545]: Error stating Not enough space available on device when publishing AIR applications to Android or iOS devices.
• [Reference #3324491]: *Windows 7 only* Deploying AIR applications to iOS devices fails intermittently.
• [Reference #3330944]: The Adobe Application Manager window does not close soon after launching the Adobe Extension Manager.

Known issues | Flash CS5.5
This document contains the current known issues for Adobe Flash Professional CS5.5. If you have further questions that this document does not answer, see the Flash Help and Support page.

Known issues

Tools
• On Mac OS, when shortcuts for tools are started with ATOK IME on, Flash Professional CS5.5 tools don’t work on the Stage.
• Saving a CS4 file in CS5 and reopening it in CS5 sometimes results in a “Flash cannot parse this document” error. To solve this issue, update your Flash Professional installation to the most recent update.
• Flash Professional CS5.5 crashes when converting more than two symbols into a symbol. This issue usually happens as a result of the computer being out of memory. To solve this issue, free up available memory. To free up memory, try restarting the system, or closing other open applications or documents. If neither of these options work, add more RAM to the system.
• In Flash Professional CS5 & CS5.5, some JPG/JPEG files will disappear from the Stage and appear blank in the library after saving. To solve this issue, update your Flash Professional installation to the most recent update.

• When a child symbol is used in multiple symbols, existing symbol content is deleted after performing a copy and paste action. This issue only happens if the symbol is not loaded (on the stage or in library) before copy and paste. To work around this issue, load the symbol by clicking it once in the library or adding it to the stage before copying and pasting.

• Custom default color palettes cannot be saved when saving as CS5.5 format (compressed and uncompressed). To work around this issue, save as CS5 first and then resave as CS5.5.

**Code Snippets**

• You cannot type double-byte characters directly into the Code Snippets panel through IME.

**XFL**

• Flash CS5 files sometimes don’t open if they contain components in the Library that have been moved to a folder other than where originally created. (For example, the issue occurs when components are dragged from the Components panel into the Library panel).

• When using Flash CS5 and CS5.5, symbols saved in XFL format (compressed or uncompressed) are corrupted if its name has a special character. Refrain from using special characters in symbol names.

• **Windows only:** When using compressed XFL files, data loss can occur when saving as a hidden file. To solve this issue, use hidden folders instead of hidden files.

• **Windows only:** You can’t save a FLA file in a Network/Desktop folder a second time after receiving an “unable to save” error message. To solve this issue, save the file as a FLA file instead.

**AIR, AIR for Android, AIR for iOS**

• An error occurs when you publish an AIR file if a surrogate pair Unicode character appears in the path to or filename of FLA file. The error also occurs when a surrogate pair Unicode character appears in the file or OS login name.

• When you use a Japanese Device font in a TLF vertical text field, Japanese symbols appear rotated on the Android runtime and AIR2.6.

• (Windows) When you publish an AIR for iOS application, an error occurs when the path to the FLA file includes a double-byte or high-ASCII character (not including the FLA filename).

• When you use a double-byte or high-ASCII character in the filename or path to any included file of an AIR for Android FLA file, the .apk file isn’t created.

• When you using the Japanese language environment of Mac OS, Flash crashes when a high-ASCII-European character is used in the AIR for Android (.apk) output filename.

**Installation**

• Flash crashes when installed at custom location with special characters.

**Flash Builder**

• Flash Builder always targets the highest version of Flash Professional installed.

**Text Layout Framework (TLF)**

• FLA does not open due to ETX characters in TLF markup. To solve this issue, update your Flash Professional installation to the most recent update.
Fonts rendered incorrectly | Flash | Mac OSX 10.6.7

Issue
After you upgrade to Mac OS X 10.6.7, Flash CS5 or earlier doesn’t render different OpenType fonts correctly. Below are screenshots of what the font looks like when rendered correctly and incorrectly:

Correct

Incorrect

Solution
Adobe is currently working with Apple to provide a permanent solution for Mac OS X 10.6.7+. In the meantime, you can downgrade to Mac OS X 10.6.6. You can find the install at Apple’s support site.

Adobe Flash Publish keyboard shortcut (Shift+F12) is used by Mac OS X 10.4

Issue
Using the keyboard shortcut of Shift+F12 to publish from Adobe Flash causes Macintosh OS X Dashboard to launch. For example:

1. Launch Flash 8 on OS 10.4 (Tiger).
2. Create a new FLA document.
3. Press Shift+F12 to publish the document.

The Tiger Dashboard application appears and the Flash SWF is not published.

Reason
The Mac OS X Dashboard keyboard shortcut conflicts with the Publish keyboard shortcut Adobe Flash uses.

Ref. (133051)
Solution
Creating a custom Keyboard Shortcut will resolve this issue.

1. Choose Edit > Keyboard Shortcuts
2. Duplicate the existing keyboard shortcut set using the Duplicate Set button
3. Name the new Keyboard Shortcut set
4. Expand the File tree and select "Publish"
5. Select and remove the existing keyboard shortcut for Edit > Publish
6. Choose + to add a new shortcut
7. Press the desired new keyboard shortcut (for example, Command + Shift + F12)
8. Choose OK to close the Keyboard Shortcut dialog box and make the new custom set the currently assigned set.

For complete information, see "Customizing keyboard shortcuts" in Adobe Flash Help.

Flash Professional CS5 known issues and bugs

Adobe Flash Professional CS5 known issues and bugs
This document contains known issues and troubleshooting tips not covered in the Flash Professional documentation. Please refer to Adobe Support for late-breaking information and known issues for all Creative Suite 5 applications.

GB18030 support for Windows XP
In order to support the display of all characters of the Chinese standard GB18030 on Windows XP systems, Adobe recommends the installation of the Microsoft GB18030 Support Package followed by the installation of an associated Flash CS5 GB18030 workaround.

The Microsoft support package will update an XP system with, among other things, fonts and input-method-editors (IMEs) to correctly support GB18030. The support package is available as a download from the Microsoft website. (#BG061690)

The Flash CS5 GB18030 workaround installation details are described in the following technote found on the Adobe Flash support page:
http://kb2.adobe.com/cps/407/kb407212.html

© 2010 Adobe Systems Incorporated. All rights reserved.

Security Update 11.5.2 | CS5.5
This security update addresses the following issue:

- A vulnerability identified in the security bulletin http://www.adobe.com/go/apsb12-12. The vulnerability allows an attacker to take control of the affected system. If you are using Flash Professional CS5.5, applying this Security Update resolves the security vulnerability. Note: The issue was fixed in Flash Professional CS6.

The Security Update is available for download at this location.
Flash Player 10, Action Script 3: Loader.unloadAndStop

History

The Loader.unloadAndStop feature was added to Action Script 3's API to automate a process that was previously manual when using Loader.unload.

Though Loader.unload removes the child of the Loader object, the unloaded object will still run in the background until it is actually disposed of by the garbage collector. Since the garbage collector does not dispose of objects that are referred to, the "unloaded" content could perhaps never be removed (for example, event listeners could reference the object).

Example: A SWF file that has a music track is loaded into an application. Later, the SWF is unloaded using Loader.unload(). Though the SWF will be removed from the screen, the music will still be heard.

SOLUTION

Loader.unloadAndStop is a new addition to Action Script 3's API. It helps developers to correctly stop and unload loaded content using the Loader.load/loadBytes APIs.

Loader.unloadAndStop is very similar to Loader.unload. However, it adds one more step. After the "unload" event is dispatched to the LoaderInfo class, Flash Player recursively attempts to stop and clear as many objects as possible (Sounds, NetStreams, EventListeners, etc.) within the loaded SWF. This feature is especially useful when unloading unknown 3rd party content.

Example

```javascript
import flash.display.Loader
import flash.net.URLRequest
import flash.events.Event

var myLoader:Loader = new Loader();
myLoader.contentLoaderInfo.addEventListener(Event.COMPLETE, function(e:Event){
    addChild(myLoader);
});

myLoader.load(new URLRequest("item2.swf");

btnUnload.addEventListener(MouseEvent.CLICK, function(){
    // Calling this method will unload and recursively stop the content of the movie.
    myLoader.unloadAndStop();
});
```
Flash OBJECT and EMBED tag attributes

This document is the target of a link in the Flash Lite 4 Porting Guide. Before you archive or unpublish, contact Learning Resources.

This document lists the attributes of the OBJECT and EMBED tags used to publish SWF (Flash-enabled) content in HTML pages for display in web browsers. The attributes allow you to specify certain parameters that control the details of how and where Flash Player displays the SWF file in the browser.

Information about including Flash-enabled (SWF file) content within web pages is also available in the TechNote OBJECT tag syntax | Flash (tn_4150).

Required attributes

The following attributes are required within the OBJECT and EMBED tags when adding a SWF file to an HTML page. The Publish command in Flash Professional creates HTML files with the required attributes specified for you.

Note: Values in brackets and italics indicate that the developer chooses the value.

Required for both OBJECT and EMBED:

- **width** - Specifies the width of the SWF content in either pixels or percentage of browser window.
- **height** - Specifies the height of the SWF content in either pixels or percentage of browser window.

Required for OBJECT tag only:

- **classid** - Identifies the ActiveX control for the browser. See example code in OBJECT tag syntax | Flash (tn_4150) for the correct value.
- **codebase** - Identifies the location of the Flash Player ActiveX control so that the browser can automatically download it if it is not already installed. See example code in OBJECT tag syntax | Flash (tn_4150) for the correct value.
- **movie (param)** - Specifies the source location (URL) of the SWF file to load.

Required for EMBED tag only

- **src** - Specifies the source location (URL) of the SWF file to load.
- **pluginspage** - Identifies the location of the Flash Player plug-in so that the user can download it if it is not already installed.

Optional attributes

The following attributes are optional when defining the OBJECT and EMBED tags. For OBJECT, all attributes are defined in param tags unless otherwise specified:

- **id** (attribute for OBJECT tag) and **name** (attribute for EMBED tag) - SWF file identifier. Identifies the SWF file to the web browser, allowing browser scripting languages (for example, JavaScript) to reference the SWF content. For cross-browser compatibility, make sure that the id and name are set to the same value.
- **play** - Possible values: true, false. Specifies whether a timeline-based SWF file begins playing immediately on loading in the browser. If this attribute is omitted, the default value is true.
- **loop** - Possible values: true, false. Specifies whether a timeline-based SWF file repeats indefinitely or stops when it reaches the last frame. If this attribute is omitted, the default value is true.
• **menu** - Possible values: true, false. Specifies if movie playback controls are available in the Flash Player context menu.
  
  • true displays a full menu that provides expanded movie playback controls (for example, Zoom, Quality, Play, Loop, Rewind, Forward, Back).
  
  • false displays a menu that hides movie playback controls (for example, Zoom, Quality, Play, Loop, Rewind, Forward, Back). This attribute is useful for SWF content that does not rely on the Timeline, such as content controlled entirely by ActionScript. The short menu includes "Settings" and "About Flash Player" menu items.
  
• **quality** - Possible values: low, autolow, autohigh, medium, high, best. Specifies the display list Stage rendering quality. Setting the Stage.quality property via ActionScript overrides this value.
  
  • low favors playback speed over appearance and never uses anti-aliasing.
  
  • autolow emphasizes speed at first but improves appearance whenever possible. Playback begins with anti-aliasing turned off. If Flash Player detects that the processor can handle it, anti-aliasing is turned on.
  
  • autohigh emphasizes playback speed and appearance equally at first but sacrifices appearance for playback speed if necessary. Playback begins with anti-aliasing turned on. If the actual frame rate drops below the specified frame rate, anti-aliasing is turned off to improve playback speed. Use this setting to emulate the View > Antialias setting in Flash Professional.
  
  • medium applies some anti-aliasing and does not smooth bitmaps. It produces a better quality than the Low setting, but lower quality than the High setting.
  
  • high favors appearance over playback speed and always applies anti-aliasing. If the movie does not contain animation, bitmaps are smoothed; if the movie has animation, bitmaps are not smoothed.
  
  • best provides the best display quality and does not consider playback speed. All output is anti-aliased and all bitmaps are smoothed.
  
• **scale** - Possible values: showall, noborder, exactfit, noscale. Specifies how Flash Player scales SWF content to fit the pixel area specified by the OBJECT or EMBED tag.
  
  • default (Show all) makes the entire SWF file visible in the specified area without distortion, while maintaining the original aspect ratio of the movie. Borders can appear on two sides of the movie.
  
  • noborder scales the SWF file to fill the specified area, while maintaining the original aspect ratio of the file. Flash Player can crop the content, but no distortion occurs.
  
  • exactfit makes the entire SWF file visible in the specified area without trying to preserve the original aspect ratio. Distortion can occur.
  
  • noscale prevents the SWF file from scaling to fit the area of the OBJECT or EMBED tag. Cropping can occur.
  
• **align** (attribute for Object) - Possible values: l, t, r.
  
  • Default centers the movie in the browser window and crops edges if the browser window is smaller than the movie.
  
  • l (left), r (right), and t (top) align the movie along the corresponding edge of the browser window and crop the remaining three sides as needed.
  
• **salign** - Possible values: l, t, r, tl, tr.
  
  • l, r, and t align the movie along the left, right, or top edge of the browser window and crop the remaining sides as needed.
  
  • tl and tr align the movie to the upper-left and top upper-corner of the browser window and crop the bottom and remaining side as necessary.
• **wmode** - Possible values: window, direct, opaque, transparent, gpu. Sets the Window Mode property of the SWF file for transparency, layering, positioning, and rendering in the browser. If this attribute is omitted, the default value is "window". For more information, see Using Window Mode (wmode) values below.
  - window - The SWF content plays in its own rectangle ("window") on a web page. The browser determines how the SWF content is layered against other HTML elements. With this value, you cannot explicitly specify if SWF content appears above or below other HTML elements on the page.
  - direct - Use direct to path rendering. This attribute bypasses compositing in the screen buffer and renders the SWF content directly to the screen. This wmode value is recommended to provide the best performance for content playback. It enables hardware accelerated presentation of SWF content that uses Stage Video or Stage 3D.
  - opaque - The SWF content is layered together with other HTML elements on the page. The SWF file is opaque and hides everything layered behind it on the page. This option reduces playback performance compared to wmode=window or wmode=direct.
  - transparent - The SWF content is layered together with other HTML elements on the page. The SWF file background color (Stage color) is transparent. HTML elements beneath the SWF file are visible through any transparent areas of the SWF, with alpha blending. This option reduces playback performance compared to wmode=window or wmode=direct.
  - gpu - Use additional hardware acceleration on some Internet-connected TVs and mobile devices. In contrast to other wmode values, pixel fidelity for display list graphics is not guaranteed. Otherwise, this value is similar to wmode=direct.

• **bgcolor** - [hexadecimal RGB value] in the format #RRGGBB. Specifies the background color of the SWF content. Use this attribute to override the background color (Stage color) setting specified in the SWF file. (This attribute does not affect the background color of the HTML page.)

• **base** - [base directory] or [URL]. Specifies the base directory or URL used to resolve all relative path statements in the SWF file. This attribute is helpful when your SWF file is kept in a different directory from your other files.

• **allowFullScreen** - Possible values: true or false. Setting this value to true allows the SWF file to enter full screen mode via ActionScript. For more information, see Exploring full screen mode in Flash Player. If this attribute is omitted, the default value is false.

• **fullScreenAspectRatio** - Possible values: portrait or landscape. Used to control how full screen SWF content appears on mobile devices that support automatic screen rotation, such as phones and tablets. If this attribute is specified, Flash Player uses the specified screen orientation (portrait or landscape) when the SWF is viewed in full screen mode. It doesn’t matter how the device is oriented. If this attribute is not specified, orientation of content in full screen mode follows the screen orientation used by the browser.

• **flashvars** - Variables, defined as a string of key=value pairs, that are passed to the SWF file.
  - Use flashvars to specify root-level variables in the SWF file. The format of the string is a set of key=value combinations separated by the ‘&’ character.
  - Browsers support string sizes of up to 64 KB (65535 bytes) in length.
  - For more information on using flashvars, see Using FlashVars to pass variables to a SWF (tn_16417).

• **browserzoom** - The following OBJECT and EMBED tags are available in Flash Player to change the the size of Flash content in response to change in browser zoom factor:
  - scale: turn on the browser zoom factor behavior
  - noscale: turn off the browser zoom factor behavior
Using Window Mode (wmode) values

Developers can set the Window Mode (wmode) property of the SWF content to control layering and transparency of the content in the browser. Regardless of the wmode value, Flash Player displays content viewed in full screen mode using direct path rendering. Using direct path rendering enables hardware accelerated presentation capabilities such as Stage Video and Stage 3D.

To deliver the highest performance playback of Flash-enabled content in the browser, use wmode=direct. This attribute enables hardware accelerated presentation capabilities, including Stage Video and Stage 3D. Avoid overlapping the SWF content with HTML elements (for example, HTML-based pop-up menus) intended to appear above the SWF file. Explicit layering control is only supported with some modern browsers (see table below). In other browsers, the SWF content always appears above other HTML elements.

If you require explicit layering control, transparency, or HTML elements that float above SWF content, use wmode=opaque or wmode=transparent. You can then control layering in relationship to other elements on the page through HTML. However, these “windowless” modes reduce playback performance compared to wmode=window (the default) and wmode=direct.

<table>
<thead>
<tr>
<th>Use case</th>
<th>Recommended wmode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best performance, including support for hardware accelerated Stage Video and Stage 3D.</td>
<td>wmode=direct</td>
</tr>
<tr>
<td>Requires overlapping HTML elements to appear on top of SWF content, transparency, or explicit layering control relative to HTML elements (across all browsers).</td>
<td>wmode=opaque or wmode=transparent</td>
</tr>
</tbody>
</table>

Browser support for Window Mode (wmode) values

Different browsers rely on different approaches for rendering web page content and handles wmode values differently. The table above summarizes when to use different wmodes. The matrix below details the specific behaviors you can expect in each browser.
<table>
<thead>
<tr>
<th>Wmode</th>
<th>Benefits</th>
<th>IE 6, 7, and 8 Win</th>
<th>IE 9 Win</th>
<th>Firefox Win</th>
<th>Chrome Win</th>
<th>Firefox 3 Mac (OSX 10.5 and later)</th>
<th>Firefox 4 Mac (OSX 10.6 and later)</th>
<th>Safari 3 (OSX 10.5)</th>
<th>Safari 4 (OSX 10.6 and later)</th>
<th>Chrome Mac (OSX 10.5)</th>
<th>Chrome Mac (OSX 10.6 and later)</th>
</tr>
</thead>
<tbody>
<tr>
<td>window</td>
<td>Broadest accessibility support</td>
<td>-/-A</td>
<td>-/-A</td>
<td>-/-A</td>
<td>-/-A</td>
<td>-/-A</td>
<td>-/-A</td>
<td>-/-</td>
<td>-/-</td>
<td>-/-</td>
<td>-/-</td>
</tr>
<tr>
<td>transparent</td>
<td>Transparency and alpha blending over HTML elements supported. HTML elements can overlap SWF content. Explicit layering control (all browsers)</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
<td>L/-/</td>
</tr>
</tbody>
</table>
Related Information

- **OBJECT tag syntax**
- **SWF file ignores stacking order, plays on top of DHTML layers**

### Flash keyboard shortcuts

The following is a complete list of keyboard shortcuts for Flash. Standard keyboard shortcuts, such as Command/Control C to copy and Command/Control V to paste, are omitted.

**Note:** Flash 5 and later support customizable keyboard shortcuts. Choose Edit > Keyboard Shortcuts to open the Keyboard Shortcuts dialog box.

<table>
<thead>
<tr>
<th>opaque</th>
<th>HTML layering is supported. HTML elements can overlap SWF content. Supports explicit control of layering in relation to other HTML elements.</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
<th>L/-/-</th>
</tr>
</thead>
<tbody>
<tr>
<td>direct</td>
<td>Best performance. Hardware accelerated presentation, including Stage Video and Stage 3D (all browsers).</td>
<td>-/H/A</td>
<td>L/H/-</td>
<td>-/H/A</td>
<td>-/H/A</td>
<td>L/H/-</td>
<td>-/H/A</td>
<td>L/H/-</td>
<td>-/H/A</td>
<td>L/H/-</td>
</tr>
<tr>
<td>gpu</td>
<td>Additional hardware acceleration on some Internet connected TVs and mobile devices.</td>
<td>-/H/A</td>
<td>L/H/-</td>
<td>-/H/A</td>
<td>-/H/A</td>
<td>L/H/-</td>
<td>-/H/A</td>
<td>L/H/-</td>
<td>-/H/A</td>
<td>L/H/-</td>
</tr>
</tbody>
</table>

L = HTML layering is supported. HTML elements can overlap SWF content. Supports explicit control of layering in relation to other HTML elements. H = Hardware accelerated presentation is supported, including Stage Video and Stage 3D. A = Accessibility is supported.
Tools
A - Arrow/Cursor
T - Text Tool
P - Pencil
I - Ink Bottle
B - Paint Brush
U - Paint Bucket
E - Eraser
D - Dropper
M - Magnifier
L - Lasso

Frames
F5 - Add frame
Shift F5 - Delete Frame
F6 - Add Keyframe
F7 - Add Blank Keyframe
F8 - Make Symbol

Timeline
Enter - Play
Command/Control 0 (zero) - Rewind
< - Previous Frame
> - Next Frame
Home - Goto First Scene
End - Goto Last Scene
Page Up - Goto Previous Scene
Page Down - Goto Next Scene

File
Command/Control R - Import Image/Sound/etc...
Command/Control Shift R - Export to .swf/.spl/.gif/, and so on
Command/Control Shift O - Open as Library

View
Command/Control 1 - View movie at 100% size
Command/Control 2 - Show Frame
Command/Control 3 - Show All

Windows
Command/Control L - Show/Hide Library
Command/Control M - Modify Movie Properties
Command/Control E - Toggle between Edit Movie and Edit Symbol Mode
Command/Control Shift L - Show/Hide Timeline
Command/Control Shift W - Show/Hide Work Area

Modifying and editing
Command/Control G - Group
Command/Control U - Ungroup
Command/Control B - Break Apart
Command/Control Shift V - Paste in Place
Command/Control D - Duplicate
Command/Control A - Select All
Command/Control Shift A - Deselect All
Command/Control Shift O - Optimize Curves
Command/Control K - Align Window
Command/Control Shift S - Scale and Rotate
Command/Control Shift Z - Remove Transform
Command/Control Shift Up Arrow - Move Ahead
Command/Control Down Arrow - Move Behind
Command/Control T - Modify Font
Command/Control Shift T - Modify Paragraph
Command/Control Left Arrow - Narrower letterspacing (kerning)
Command/Control Right Arrow - wider letterspacing (kerning)

Other tricks
With the arrow cursor:

(Deprecated with Flash Professional CC) Command + Drag - Duplicates a shape Press the control key first (Mac OS and Windows) and drag a selected shape or group of shapes to duplicate those shapes where you release the mouse button.

Command/Control Clicking a keyframe to move frame - Command/Control clicking a frame in the timeline switches the cursor to a slider. It allows you to drag that frame to a new place in the timeline within that same layer. This function is useful for editing the positioning of tweens, and so on, in the timeline.

With the magnifier tool:
Control + Click - Toggles to the opposite magnifier. If the + magnifier (zoom in) is active, and you hit Control while clicking, you switch to the - magnifier and actually zoom OUT.

With the dropper tool:

Shift + Click - Select a color for both fill and outline tools. Clicking a red fill does the same, giving you the Bucket tool, and switching fill colors to red. But the outline tool colors are not changed. Clicking text switches the text tool to that color, and gives you the text tool. Shift clicking with the dropper makes the color you click active for ALL tools. It doesn't automatically switch you to any tool. It leaves the dropper active.

This shortcut is one of the least known for Flash. It's the only way to use the dropper on an outline for example, and then switch to the fill tool and have that color automatically active.

Expanding/Collapsing some sections:

Windows - Space Bar or Control + Enter

MAC - Command + Enter

For example, go to Publish Settings > Flash (.swf). Advanced options can be expanded by using the above-mentioned shortcuts.

Toggle Breakpoint:

• Flash Professional CC - Command/Control + Alt + B
• Flash Professional CS6 and earlier versions - Command/Control + B

Remove All Breakpoints:

• Flash Professional CC - Command/Control + Shift + Alt + B
• Flash Professional CS6 and earlier versions - Command/Control + Shift + B

Workspace Accessibility Commands:

With Flash Professional CC, the workspace accessibility commands are listed under the Goto section in the keyboard shortcuts (Edit > Keyboard Shortcuts) panel.

Error "Unexpected element/attribute" | Publish iOS application | Flash Professional CS5.5

Issue

When you publish an iOS application in Flash Professional CS5.5, you receive an error like the one below.

Error 103: application.version is an unexpected element/attribute

Solution

1 Create an AIR For iOS FLA file.
2 Save the file.
3 Test the Movie (Control > Test Movie).
4 Browse to the file location and open the application descriptor file (XML file).
5 Switch back to Flash and open the AIR publish settings dialog (File > AIR For iOS Settings).
6 Change the version number to anything you want. For example, change it to 1.2.
7 Save the file and test the movie (Control > Test Movie).
8 Switch back to the application descriptor file.
9 Find any instance of the <version> tag (Edit > Find And Replace) and delete them all.
10 Update the <versionNumber> tag in the XML file by adding the new version you want as three-digit number. For example: <versionNumber>1.2.0</versionNumber>
11 Save and publish the file.

Error "Could not find an Android device..." | Publish Android app | Flash CS5, CS5.5

Issue
When you publish an AIR for Android application in Flash CS5 or CS5.5, you receive the following message:

Mac OS: "Could not find an Android device to connect to. Please ensure that you have a device connected, and the device has USB debugging enabled."

Windows: "Could not find an Android device to connect to. Please ensure that you have the phone connected, the correct drivers installed, and the phone has USB device debugging enabled."

You selected the option to install the application on the connected device.

Solutions

Solution 1: Try the following general troubleshooting techniques:
- Unplug the device from the USB port and plug it back in.
- When the USB connection on the device is set to PC Mode, the device is not recognized when plugged in to USB. Set the USB connection to Charge Only mode. On the device, pull down the status menu, select USB connection and set to Charge Only.
- Turn USB Debugging off and on by following these steps.
  1 Select the Home button on the device.
  2 Press the Options button on the device and select the Settings option.
  3 Navigate to the Applications > Development option and find the USB Debugging preference.
  4 Turn USB debugging ON and then OFF.

Solution 2: Ensure that the correct driver is installed for your device.
Flash Professional CS5 and CS5.5 include a series of drivers for some of the most common devices. However, this list is not a complete list of drivers for all known devices.
**Note:** These drivers are for Windows only. If you are running a Mac, it's unnecessary to install any drivers, as the general troubleshooting techniques above should work. If your device is not found using the process below, visit your device manufacturer's website to obtain the correct drivers.

Follow the instructions below to find and install the driver for your device:

**Windows 7**

1. Connect your Android device to your computer's USB port.
2. Windows tries to install the driver software. If it fails with a "No Driver Found" error, proceed to the next step.
3. Right-click Computer on the desktop or Windows Explorer, and select Manage.
4. Select Device Manager in the left pane of the Computer Management window.
5. Locate Other devices and right-click your device and select Update Driver Software.
6. A prompt asks how you want to search for the driver software. Select Browse My Computer For Driver Software.
7. Click Browse and browse to `<Flash Installation Folder>\AIR2.6\install\android\usb_drivers\`. Select Include Subfolders.
8. Click Next.
9. When Windows 7 asks if you'd like to install the device software, click Install to install the driver.

**Windows Vista**

1. Connect your Android-powered device to your computer’s USB port. Windows detects the device and launches the Found New Hardware wizard.
2. Select Install From a List Or A Specific Location (Advanced) and click Next.
3. Select Include This location In The Search and click Browse and navigate to `<Flash Installation Folder>\AIR2.6\install\android\usb_drivers\`.
4. Click Next. If Vista prompts you to confirm the privilege elevation required for driver installation, confirm it to install the driver.

**Windows XP**

1. Connect your Android-powered device to your computer’s USB port. Windows detects the device and launches the Found New Hardware wizard.
2. When Windows XP asks to use Windows update to search for the driver, select No, Not This Time and click Next.
3. Select Install From a List Or Specific Location (Advanced) and click Next.
4. Select Include This Location In The Search and click Browse. Navigate to `<Flash Installation Folder>\AIR2.6\install\android\usb_drivers\`.
5. Click Next to install the driver.

**Verify that the device is connected properly**

1. Plug the device to the computer via USB.
2. Launch the command window on Windows (Start menu, type cmd in the Search for programs or files field), or Terminal on Mac OS (/Applications/Utilities/Terminal).
3. Run the following command: `adb devices`  
   **Note:** `adb` is installed in AIR2.6/lib/android/bin.
4 Verify that the device is listed, and that it does not say Offline (It should be listed as something like "015DC3C60101702E device"). If it’s not listed or shown as offline, try turning the USB debugging ON, or OFF and then ON (see steps above).

5 On the device, click the Home button and then choose Options > Settings > Applications > Development > USB debugging.

6 Run adb -devices again. If it still doesn’t show or show as offline, check to see if you have the driver installed. Also check to make sure that the USB connection is set to Charge only (see steps above).

How to duplicate a movie clip in random locations

To place Movie Clips in random positions on the screen, you will use the Duplicate Movie Clip, Set Variable and Set Property statements, and the Random function, all of which are new to Flash 4. This tutorial assumes that you have read the "Creating Interactivity" section of the Flash Help which can be found in either the print manual or the HTML Help.

There are at least two ways to create this effect: You can have the clips appear when the viewer hits a Button, or they can appear when a certain frame in the movie is reached especially in a loop. We’ll go through both of these.

First, a primer on the statements we’ll use:

• **Duplicate Movie Clip** creates a Movie Clip identical to one you specify. You must specify the target path of the original (How To Use Tell Target ), an instance name for the new duplicate, and a layer depth for the new duplicate.

• **Set Variable** assigns a value to a variable, whether the variable is new or preexisting. You must specify the name and value of the variable.

• **Set Property** can be used to change certain properties of a Movie Clip instance, such as X position, Y position, Visibility or Alpha value. You must specify the name of the property to be changed, the target path of the Movie Clip instance whose property is to be changed, and a new value for the property.

• **Random** is a function that generates a random number between 0 and one less than the number you specify. For example, Random (10) generates the number 0, 9, or something in between.

To place Movie Clips in random positions on the stage when the user clicks a Button, follow these steps:

1 Place your Button and Movie Clip on the stage, and give the clip an instance name by selecting the Movie Clip and choosing Modify>Instance. In this example, the spinning flower is namedspinnyn

2 Double-click on the Button to bring up the Instance Properties dialog box.

3 Select the Actions tab.

4 From the plus menu, select Duplicate/Remove Movie Clip. Fill in the dialog boxes as follows:

• **Action:** Duplicate Movie Clip

• **Target:** spinnyn (Or your Movie Clip instance’s name). If you wish, you can use the Target Editor to choose the target visually. The tab to the right of the box should be set to "String Literal."

• **New Name:** A generic Movie Clip name, such as "MC", concatenated with a variable using an ampersand. For example, "MC" & var See finished product . The tab should be set to "Expression."
• **Depth**: A variable name, in the example the variable is named var. The variable for "New Name" and "Depth" should be the same. This way we only need one variable in one Set Variable statement. The tab should be set to "Expression."

5 From the plus menu, select Set Property. Fill in the dialog boxes as follows:
   • **Set**: X Position
   • **Target**: This is the same name you specified in "New Name" above. The tab should be set to "Expression," just as with "New Name."
   • **Value**: Random(350). Thus, the X Position of the duplicate Clip is set to a random integer between 0 and 349. Random is a function, so the tab must be set to 'Expression.'

6 Repeat step 5 for the new Y Position.

7 From the plus menu, select Set Variable and fill in the dialog boxes as follows:
   • **Variable**: var (Or a variable name of your choice, as long as it is the same as the one you put in "Depth," and put on the end of the name you wrote in "New Name" in "Duplicate Movie" above.)
   • **Value**: var + 1. This increments the var variable so that the next time the Button is clicked, the next Movie Clip has a different name and is placed in a different layer.

8 Click OK.

**Finished Product**

After you've done the steps above, the action script in the Button's Instance Properties box should look like this:

```actionscript
On (Release) Duplicate Movie Clip ("/spinny", "MC"&var, var) Set Property ("MC"&var, X Position) = random(350) Set Property ("MC"&var, Y Position) = random(350) Set Variable: "var" = var+1 End On
```

**Note:** You can add as many Set Property statements as you like. In the Flower example shown above, I also included the statement below to set the Alpha property of each new Movie Clip to a value between 10 and 100:

```actionscript
Set Property ("MC"&var, Alpha) = Random(91)+10
```

**Placing Movie Clips when the Movie reaches a certain frame**

Simply follow the same instructions as above, but place the actions in the Frame Properties box by double-clicking a keyframe where the action should occur. Also, the script does not need the On (Release) and End On actions when placed in a frame.

**Note:** To create a "Clear" Button that removes all the duplicate Movie Clips from the stage, use the Loop action. The action script on the "Clear" button looks like this:

```actionscript
On (Release) Loop While (count <= var) Remove Movie Clip ("MC" & count) Set Variable: "count" = count + 1 End Loop Set Variable: "var" = 0 Set Variable: "count" = 0 End On
```

**Q/A**

**Why does the name of the duplicate clip have to be that expression, instead of something friendly?**

The expression "MC"&var evaluates as "The literal letters MC, concatenated (stuck together with) the value of the variable var." By naming the duplicate clip this way, the Button will keep creating new duplicates with new names (MC1, MC2, etc.) as long as the viewer keeps clicking the Button.

**Why does the duplicate's level have to be set as var?**

Since the value of var increments with each Button click, this ensures that each clip is put on its own layer. If the depth were set as a number, each clip would replace the previous one, since they would both occupy the same layer. Try it and see.
Why was 350 chosen as the range for Random?

It didn’t have to be, but since the Set Property statement defines X and Y position in pixels, and this Flash Player Movie is 350 pixels high and 350 pixels wide, this range ensured that the clips would be placed on the screen.

Additional information

For additional information about advanced interactivity in Flash, please refer to How To Use Tell Target (TechNote 13479).

Differences between the Equality operator (==) and the Assignment operator (=)

In Macromedia Flash 5, the assignment operator (=) and the equality operator (==) appear to be similar but are used for completely different reasons.

The assignment operator (=) is used to assign a value to a variable, element of an array, or property of an object. Here are a few examples of using the assignment operator:

```javascript
// Set a variable to a value. var userName = "Jack"; // Set an element of an array to a value. namesArray[3] = "Jack"; // Set a property of an object to a value. nameClip._visible = FALSE;
```

The equality operator (==) is used to compare two values or expressions. It is used to compare numbers, strings, Boolean values, variables, objects, arrays, or functions. The result is TRUE if the expressions are equal and FALSE otherwise. How items are compared depends on their data type:

- Numbers, strings, and Boolean values are compared by value, and are considered equal if they have the same value. For instance, two strings are equal if they have the same number of characters.

- Variables, objects, arrays, and functions are compared by reference. Two variables are equal if they refer to the same object, array, or function. Two separate arrays are never considered equal, even if they have the same number of elements.

For example, in the following expression, the value of `userName` and the string "Jack" are the same and expression evaluates to TRUE:

```javascript
(userName == "Jack"); // TRUE. // However... (userName == "jack"); // FALSE.
```

The second expression evaluates to FALSE because "Jack" is not the same string value as "jack".

However, when comparing two variables, array, objects, or functions, an equality expression will evaluate to TRUE only if the items being compared point to the same reference. For example, consider the following:

```javascript
var userName = "Jack"; var user = "Jack";
```

Although the two variables contain the same value the variables themselves are not equal, since they point to different references. For example:

```javascript
(userName == user); // FALSE // However... (eval(userName) == eval(user)); // TRUE
```

The second expression is TRUE because the two variables were first evaluated (using the eval function) and their values, rather than their references, are being compared.

A practical example of using the equality operator

The equality operator is often used in the following manner to check if a movie has completely loaded:

```javascript
if (_framesloaded == _totalframes) { gotoAndPlay (3); } else { gotoAndPlay (1); }
```
Additional Information

For more details on preloading a movie please refer to How to create movies that download before playing (TechNote12588).

Creating a mask that follows a motion guide

Note: The information in this TechNote is specific to Macromedia Flash 5 and earlier. Macromedia Flash MX adds the ability to use animated or scripted movie clips as masks. The techniques in this technote will continue to work in Macromedia Flash MX, however it would be simpler to use the new features of Macromedia Flash MX. For complete details refer to the Macromedia Flash MX documentation.

Motion Tweening Masks in Flash 5 and earlier versions

Many interesting effects can be created using masks together with motion tweens. This is a great way to get an animated mask. Some users want the tweened mask to follow a motion guide. But since a layer can't be both a mask layer and a guide layer, this is not possible using a single SWF.

However, this effect can be achieved using two Flash movies. In one movie, a motion tween is created that follows a guide layer. This movie is then exported as an SWF and imported into the second movie. In the second movie, the layer that contains the imported SWF is set to be a mask layer. Finally, the object to be masked is placed in a masked layer under the imported tween.

Prerequisites: This tutorial assumes that you are familiar with the basic concepts of masks, as discussed in Creating masks and creating motion guides, as covered in How to add and use a motion guide (TechNote 14133).

Download the source files for this example mask_follow_guide.zip (17K)

To have a mask follow a motion guide do the following:

1 In the first movie, create a motion tween that follows a guide layer. This animation will be used as the mask in the second movie. The shape of the tweened object needs to be the shape that you want the mask to be.

2 Publish, or export, the first movie as an SWF.

3 Import the SWF exported above into another Flash movie using the File > Import command. This will bring in the SWF as a frame by frame animation on one layer, which will be used as the mask layer.

4 Change the imported layer's property to Mask and add the material that you want masked to a layer below the mask layer.

Additional Information

If your motion tween does not follow the guide layers see, Motion path animation doesn't follow path (TechNote 4104). If you encounter problems with using a movie clip in the mask layer see Problems using movie clips in mask layers (TechNote 14264).

Creating advanced buttons

This document introduces you to more design possibilities in the construction of Flash Buttons. For more resources on performing specific actions with buttons, see Additional Information at the bottom of this document.

Prerequisites
To get the most out of this TechNote, an understanding of creating basic buttons is required. We recommend doing the “Buttons” Lesson within Flash (Available from the Help menu). More button resources are available in Additional Information at the bottom of this document.

Contents

- Button Hit states
- Invisible buttons
- Buttons with "animated" states
- Rollover in one area of the movie affects another area of the movie
- Example
- Additional Information

Button Hit states

Hit states are important to understand when designing a complex button. Its shape and area represent the active area of the button. Become familiar and comfortable with the Hit state.

Experimenting with the Hit state

1. Introduce a button from the Common Libraries onto the Stage. Edit the button.
2. Highlight the frame in the button's Hit state, and insert a Key frame. Test this behavior in a movie.
3. Resize the object in the Hit state dramatically, and test.
4. Delete the Keyframe in the Hit State. Test this.
5. Notice how these changes affect your button cursor, and the Up, Over and Down states that you see when the button is enabled.

Invisible buttons

The Up, Over and Down states of a button may be left empty. If these states are empty, the Hit state must then be defined (contain content).

When the Up state of the button is empty or 'invisible', the button is represented on the Stage by a blue shape equivalent to the content in the next Keyframe within the button. This blue shape will not be visible in your final movie. See the Example to see how this works.

Buttons with "animated" states

To make animated buttons in Flash, place a Movie Clip in the button state that is to be animated.

Create an animated button

1. Create a movie clip for each state of the button that you want animated.
2. Create the button.
3. Place the movie clips in the button states) to be animated.
4. Place the button on the stage.

See the Example to see how this works.

Note: Movie clips cannot be tested in the Flash editor. You will need to do a Control > Test Movie or export it as a .swf file to test.

Rollover in one area of the movie affects another area of the movie
To create this effect, move the Hit state elements to a different area of the Stage than elements in the Over state of the button. This will work for simple rollover effects. The active area of the button, will then be located in a different area onstage than the rollover effect. See the Example to see how this works

**Example**

**Note:** The Macromedia Flash Player is required to see this example.

Download Windows source file advanced_buttons.zip (17 K)
Download Macintosh source file advanced_buttons.sit (18 K)

**Note:** Flash 4 or later is required to open the source file.

**Additional information**

For more information on basic buttons, see the Using Flash manual.

To learn how to assign an action to a Button, do the Buttons lesson inside Flash. See the Using Flash manual for more information on Mouse Events and assigning actions to Buttons. See the ActionScript Reference for more advanced information on what actions can be performed.

To create a simple animated menu, see **How to create a pop-up menu in Flash** (TechNote 14363).

To create a link to an e-mail address, see **How to create a button that will send e-mail messages** (TechNote 14072).

To make a button for clearing fields in a Flash form, see **How to create a reset button** (TechNote 15541).

---

**Create a simple sound toggle | Flash**

This TechNote explains how to create a simple sound toggle control like the one shown in the sample movie below (requires **Flash Player 5 or later**). The basic technique involves inserting a streaming sound inside a movie clip. The sound can be controlled with simple ActionScript 2.0 Play and Stop actions attached to buttons. Because the sound is set to Stream, the sound starts up where it was last stopped.

Before using this TechNote, be sure that you are working with a FLA file with the Script setting set to ActionScript 2.0 in the Document properties.

**Create the sound movie clip**

1. Choose Insert > New Symbol and choose Movie Clip as the symbol’s behavior. Give the movie clip a name of your choosing.
2. Choose Import from the File menu and browse to the folder containing the sound file you want to play and import it into the movie’s Library.
3. In the first frame of the movie clip, insert add a Stop action.
4. Select frame 2 and choose Insert > Blank Keyframe. Then drag the imported sound symbol from the Library (Window > Library) to the stage of the movie clip.
5. Add the frame label “start” where the sound begins in frame 2.
With frame 2 still selected, open the Sound panel (Window > Panels > Sound), and choose Stream from the Sync pop-up menu. **Note:** Event sounds don’t work with this method.

Insert frames into the movie clip’s timeline until the waveform for the sound disappears, as shown below.

At the frame where the waveform ends, insert a keyframe and add the following action to the frame: `gotoAndPlay("start");` This action causes the playhead to return to the beginning of the sound when it’s complete. If you’d prefer the sound to not loop, insert a Stop action on this frame rather than the `gotoAndPlay` action.

Return to Edit Movie mode (Edit > Edit Movie) and drag an instance of this clip from the Library to the stage. Give the clip an instance name of “mc” in the Instance panel (Window > Panels > Instance).

**Add buttons to control the sound clip**

Add two buttons to the stage to stop and start the sound. This example uses the gel Right and gel Stop buttons from the Buttons common Library (Window > Common Libraries > Buttons > (circle) VCR Button Set).

Select the button you want to cause the sound to play and open the Actions panel (Window > Actions). Add the following ActionScript to the button:

```actionscript
on (release) {
    with (mc) {
        play();
    }
}
```

This step causes the movie clip containing the sound to start playing, which starts the sound.

Select the button you want to cause the sound to stop. Open the Actions panel (Window > Actions). Add the following ActionScript to the button:

```actionscript
on (release) {
    with (mc) {
        stop();
    }
}
```

This button causes the movie clip containing the sound to stop playing, which stops the sound.

Test the movie by choosing Control > Test Movie. If the sound doesn’t start or stop on command, make sure that the instance name you’ve assigned to the sound clip is "mc." Also make sure that the sound’s Sync property in the Sound panel is set to Stream and not Event.

**Create pop-up menus | Flash**

This TechNote walks you through making a pop-up menu in Flash, similar to the one shown in the example movie below.
The menu smart clip that ships with Macromedia Flash 5 provides another way to add pop-up menus to Flash movies. See Customizing the Flash 5 Menu Smart Clip (TechNote 15140) for details.

Note: Movie shown in small scale.

Download Windows source file popup_menu.fla.zip (8K)
Download Macintosh source file popup_menu.fla.sea.hqx (8K)

Make the movie
Follow all of the steps exactly, and refer to the images when necessary. You may wish to print out these instructions for easy reference.

How this movie functions:
1 Each Menu has its own segment of the movie’s timeline. This avoids conflicts between any submenu items.
2 The rollover actions are performed by invisible buttons. Each menu item has its own invisible rollover button.
3 The rollover button is a large button the size of the entire stage area with a cutout for the menu items.
4 The menu items and hidden buttons cannot overlap or they may not function.

The steps are broken down into the following sections:

Create layers
1 Create four layers, by choosing Insert > Layer from the Flash menu until you have a total of four layers.
2 Name the top layer "Actions/Labels".
3 Name the second layer from the top "Top level buttons".
4 Name the third layer from the top "Submenu buttons".
5 Name the bottom layer "Invisible buttons".

Set up layer: Actions/Labels:
1 Highlight frame 1 of the Actions/Labels layer.
2 Choose Modify > Frame.
3 In the Label tab, label the frame "START" (without quotes).
4 Select the Actions tab and assign a Stop action by clicking the plus "+" sign and select "Stop". Click OK.
5 Highlight frame 10 of the Actions/Labels layer.
6 Choose Insert > Keyframe (or use F6 on your keyboard) to insert a keyframe.
7 Choose Modify > Frame.
8 In the Label tab, label the frame "MENU 1".
9 In the Actions tab, assign a "Stop" action. Click OK.
10 Highlight frame 20 the Actions/Labels layer and press F6 to insert a keyframe.
11 With this frame highlighted, select Modify > Frame.
In the Label tab, label the frame "MENU 2".

In the Actions tab, assign a "Stop" action. Click OK.

Highlight frame 30 of the Actions/Labels layer and press F6 to insert a keyframe.

Lock this layer by clicking the padlock icon.

Create the menu's button:

1. Choose "Insert > New Symbol". Name it "Button", choosing Button as the behavior. Click OK.
2. Choose the Rectangle tool, and select a medium green-colored fill.
3. In the center of the stage, draw a rectangle similar to the one in the example shown, about 50 pixels high by 200 pixels wide.
4. Highlight the "Over" frame, and press F6 to insert a keyframe here.
5. Choose the Paint Bucket tool and select a dark green color from the palette.
6. Apply the Paint Bucket to the rectangle, changing it to dark green. This will make the button turn dark green while the cursor is over it, in the finished movie.
7. Highlight the "Down" Frame, and press F6 to insert a keyframe.
8. Choose light green as the Paint Bucket color, and apply it to the green rectangle. This will make the button turn light green while the button is pressed in the finished movie.
9. Choose Edit > Edit Movie to return to the main timeline.

Create the Top Level menu:

1. Open this movie's Library by choosing Window > Library.
2. Find "Button" in the Library, and drag a Button from the Library window to the upper left of the stage. Drag from the thumbnail preview of the Button in the window.
3. Drag a second instance of Button from the Library window to the upper right of the stage.
4. Line the two buttons up across the top of the stage, about 100 pixels apart.
5. Choose the Text Tool. For the font, choose Arial, 24 point, White, Bold.
6. Create a text block next to the first button that says "MENU 1".
7. Do the same as above for the second button, and type "MENU 2".
8. Using the Arrow tool, select and center the text on the buttons.
9. Highlight the MENU 1 button on the Stage.
10. Choose Modify > Instance. In the Actions tab, click the "+" sign, and choose "OnMouseEvent" and on the right, choose "Roll Over".
11. Click the "+" again, and choose "Goto". On the right, choose "Label", selecting "MENU 1" from the pop-up menu. Click OK.
12. Highlight the MENU 2 button on the Stage.
13. Choose Modify > Instance. In the Actions tab, click the "+" sign, choosing "OnMouseEvent" and on the right, choose "Roll Over".
Click the "+" again, and choose "Goto". On the right, choose "Label", selecting "MENU 2" from the pop-up menu. Click OK.

Select frame 20 of the Top level buttons layer and press F6 to insert a keyframe.

Lock this layer.

Create the Submenu items

1. Highlight frame 10 of the Submenu buttons layer and press F6 to insert a keyframe.
2. From the Library window, drag an instance of Button on the stage below MENU 1.
3. Using the Arrow tool, center it about 20 pixels below the MENU 1 button.
4. Drag another instance of the Button on the stage, centering it about 20 pixels below the last button.
5. Choose the Text Tool.
6. On the button below MENU 1, create type that says "SUBMENU 1a".
7. On the button below SUBMENU 1a, create type that says "SUBMENU 1b".
8. Use the Arrow tool to center the text on the buttons.
9. Highlight frame 20 of the Submenu Buttons layer.
10. With frame 20 selected, choose Insert > Blank Keyframe, or press F7 on your keyboard. **Note:** You must insert a BLANK keyframe here or the example will not work.
11. Drag an instance of the Button from the Library on the stage below MENU 2.
12. Using the Arrow tool, center it about 20 pixels below the MENU 2 button.
13. Drag another instance of the Button on the stage, centering it about 20 pixels below the last button.
14. Choose the Text tool.
15. On the button below MENU 2, create type that says "SUBMENU 2a".
16. On the button below SUBMENU 2a, create type that says "SUBMENU 2b".
17. Use the Arrow tool to center the text on the buttons.
18. Lock this Layer.

Creating the invisible buttons

Invisible button 1:

1. Highlight frame 10 of the Invisible buttons layer.
2. Press F6 to insert a keyframe.
3. Select the Rectangle tool, and choose a red fill. **Note:** The color is not important. This example uses red for visibility and contrast.
4. Draw a rectangle covering the entire stage, slightly larger than the stage.
5. Now select yellow as the fill color. **Note:** The color is not important. This example uses yellow for visibility and contrast.
6 Draw a rectangle around all three MENU 1 buttons. Make sure all three buttons are completely surrounded by leaving a yellow border around them.

7 Using the Arrow tool, select the yellow area. Double-click the filled area so that both the fill and outline are selected and delete using the Delete key.

8 The yellow area should be removed, revealing the background. Now the red shape covers everything except the Menu 1 items, and will become the rollover area.

9 With the Arrow tool, select the red area.

10 Choose Insert > Convert to Symbol.

11 Name this symbol "Invisible Button 1" and choose "Button" as its behavior. Click OK.

12 Choose Edit > Edit Symbols.

13 Highlight the "Hit" frame in the Button's timeline and press F6 to insert a keyframe. This copies the red area from the "Up" frame into the "Hit" frame.

14 Highlight the "Up" frame and select the red area with the Arrow tool.

15 Press Delete. This removes the red area from the "Up" state. This button now only has content in the "Hit" state, making it an invisible button.

16 Choose Edit > Edit Movie. The previously red button is now a transparent blue. This represents the invisible button you just created. **Note:** Flash assigns this light blue color to buttons that are "invisible". Invisible buttons don't have an "Up" state, but do have a "Hit" state. Flash shows this color in the Flash editor, but renders this invisible in the finished movie.

17 Click this transparent blue invisible button to highlight it.

18 Choose Modify > Instance.

19 In the Actions tab, click the "+" sign, choosing "OnMouseEvent". On the right, choose "Roll Over".

20 Click the "+" sign again and choose "Goto".

21 On the right, choose "Label", selecting "START" from the pop-up menu. Click OK.

**Invisible Button 2:**

1 Highlight frame 20 of the Invisible buttons layer.

2 Choose Insert > Blank Keyframe. **Note:** This must be a BLANK Keyframe for the example to work.

3 Choose the Rectangle tool, with a red fill color.

4 Draw a rectangle covering the entire stage, slightly larger than the stage.

5 Now, select yellow as the fill color and draw a rectangle around all three MENU 2 buttons. Make sure all three buttons are completely surrounded by leaving a yellow border around them.

6 Using the Arrow tool, select the yellow area and delete it. The yellow area should be removed, revealing the background.

7 With the Arrow tool, select the red area. Double-click to select the line surrounding it as well.
Choose Insert > Convert to Symbol.
Name this symbol "Invisible Button 2" and choose "Button" as its behavior. Click OK.
Choose Edit > Edit Symbols.
Highlight the "Hit" frame in the button's timeline and press F6 to insert a keyframe.
Highlight the "Up" frame.
Select the red area with the Arrow tool. Double-click if necessary to select both the fill and the line surrounding it and delete. This removes the red area from the "Up" state.
Choose Edit > Edit Movie. The previously red button is now a transparent blue. This represents the hidden button you just created.
Click this transparent blue invisible button to highlight it.
Choose Modify > Instance.
In the Actions tab, click the "+" sign, and choose "OnMouseEvent". On the right, choose "Roll Over".
Click the "+" sign again and choose "Goto".
On the right, choose "Label", selecting "START" from the pop-up menu. Click OK. The movie is now complete.
Select Control > Test Movie view the finished product. If you have problems, open a new document and try again, following the steps exactly.

Additional information
There are many ways to create pop-up menus. An example of a Hierarchical Menu is also available from the Flash menu, under Samples > Hierarchical.
For tips on troubleshooting a Flash movie, please refer to How to Troubleshoot a Flash Movie (TechNote 14218).

How to create password verification
Several features in Macromedia Flash can be used to set up a password checking system. Using editable text fields, variables, and the conditional If statement, a mechanism can be created to compare a password to a predefined string. The result is evaluated and actions based on whether or not a user has entered the correct password can be executed.

The procedure outlined in this TechNote does not provide a high level of security. Client-side password verification is subject to hacking and sabotage, and cannot be truly secure. It is still possible for someone to intercept a password or break through your scheme, and Flash movies are no exception to this. For truly secure password verification use server-side authentication.

Note: The steps in this TechNote make use of the Push Button and TextField components. These components belong to Macromedia Flash UI components set 2 and set 5, respectively. To download these UI components go to the Macromedia Exchange.

There are three main steps to building this sample:

- Collect the data
- Compare the user’s entry against the correct password
- Decide what action to take based on password result
Collect the Data

1. Open a new movie.
2. Navigate to Windows > Component and drag a TextField component from the Component window found in Flash UI components Set 5.
3. In the Property inspector name the instance of the TextField component myText.
4. Select the Parameters tab in the Property inspector. For Type choose Input. Select True for Password and False for Multiline parameters. Leave other parameters as default values.

Compare the Data

1. Drag a Push Button component from the UI components Set 2 and type a name for Label and click handler. For this example, Submit is used for Click Handler.
2. Create a new layer named Actions. This is where all your actions should reside in your movie. Select the first frame of the Actions layer, open the Actions panels in Normal mode.
3. Double-click the Stop action found under the Movie Control folder inside the Actions folder. This will prevent the movie to from playing automatically when published or exported.
4. Under the Actions, choose Function inside the User-Defined Functions folder. Type Submit in the Name text field.
5. Double-click the If action found in the Conditions/Loops folder (Actions > Conditions/Loops > If).
6. In the Conditions field, enter myText.text == "secret". This compares the value of password, which the user has entered in the TextField component, with the string "secret." This string is the correct password, which can be defined by a user.

Send the User to the correct location

1. Double-click the Goto action (Actions > Movie Control > Goto). Select the Goto And Stop action. Enter 3 in the Frame text field. This will be the action that is executed if the condition is true and will provide confirmation that the password is correct.
2. Double-click the Else statement (Actions > Conditions/Loops > Else). Assign a Goto action in the following statement. Enter 2 in the Frame input text field. Select the option Go to and Stop. This action will be executed if the password entered by the user is incorrect.
3. Insert a blank keyframe in Frame 2.
4. Create or drag a button from the Library panel to the stage.
5. Select the button and open the Actions panel.
6. Double-click the Goto action. This action will take the user to the first frame of the movie if password entered is incorrect.
7. Insert a blank keyframe in Frame 3.
8. Click the Text tool and make sure that Static text is selected as the text type in the Property inspector. Type a confirmation message that tells the user they have successfully entered the password.
How to create a color fade effect in Flash

**Tweening color effects** A fade-out effect is achieved in Flash by creating a Motion Tween, and adjusting the Alpha property of a Symbol. A simple example of creating this effect is provided below. The Symbol created will fade out over the duration of ten frames. The instructions provided are written for Flash 5. However, if you are using a previous version of Flash, the basic method is the same.

This process can be applied to any Symbol, including those containing text. Make sure that the object has been converted to a Symbol before applying a Motion Tween.

**A basic example**

To fade a blue circle to white over ten frames, do the following:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Open a new document, with a white background.</td>
</tr>
<tr>
<td>2</td>
<td>Select the Circle Tool, using a blue color. (Or any other desired color).</td>
</tr>
<tr>
<td>3</td>
<td>Draw a circle on the Stage.</td>
</tr>
<tr>
<td>4</td>
<td>Double-click the circle to highlight both the line and fill.</td>
</tr>
<tr>
<td>5</td>
<td>Choose Insert &gt; Convert to Symbol. Select “Graphic” and click OK.</td>
</tr>
<tr>
<td>6</td>
<td>Select frame 10 in the Timeline. Choose Insert &gt; Keyframe.</td>
</tr>
<tr>
<td>7</td>
<td>Highlight the Symbol on the Stage. Select Modify &gt; Instance.</td>
</tr>
<tr>
<td>8</td>
<td>Choose the Effects Panel. <strong>Note:</strong> In previous versions, this is called the “Color Effect” panel.</td>
</tr>
<tr>
<td>9</td>
<td>Choose “Alpha” from the pop-up menu.</td>
</tr>
<tr>
<td>10</td>
<td>Enter the number “0” as the percent. Hit the Enter key on your keyboard to activate the change.</td>
</tr>
<tr>
<td>11</td>
<td>Highlight frame 1 and choose Modify &gt; Frame, to access the Frame panel.</td>
</tr>
<tr>
<td>12</td>
<td>In the Tweening dialog box, choose “Motion” from the menu. <strong>Note:</strong> In previous versions, this can be found as a Tweening tab. Hit the Enter key to activate the change.</td>
</tr>
<tr>
<td>13</td>
<td>Choose Control&gt; Test Movie to see the fade effect.</td>
</tr>
</tbody>
</table>

**Create buttons | Flash**

Buttons are symbols that contain four frames. Each frame of a button symbol represents a different state for the button: Up, Over, Down, and Hit. These states determine how a button visually behaves when the mouse is rolled over it or when the user clicks the button. This document explains how to create basic and advanced buttons.
Create a basic button

1. Choose Insert > New Symbol, or press Control+F8 (Windows) or Command+F8 (Mac OS).
   **Note:** In Flash 3 and earlier, deselect everything on the stage and choose Insert > Create Symbol.

2. In the Symbol Properties dialog box, enter a name for the new button symbol and choose Button as the Behavior option. Click OK. Flash switches to symbol-editing mode. The Timeline header changes to display four consecutive frames labeled Up, Over, Down, and Hit. The first frame, Up, is a blank keyframe.

3. To create the Up state button image, use the drawing tools, import a graphic, or place an instance of another symbol on the Stage. You can use either a movie clip or graphic symbol in a button. However, you cannot use another button in a button. Use movie clip symbols if you want to create an animated button.

4. Select the second frame, labeled Over, and choose Insert > Keyframe. The button image from the first frame appears on the Stage.

5. Change the button image for the Over state. Repeat steps 4 and 5 for the Down frame and the Hit frame. **Note:** The Hit frame is not visible on the Stage on playback, but it defines the area of the button that responds when clicked. Make sure that the Hit frame graphic is a solid area large enough to encompass all the graphic elements of the Up, Down, and Over frames. It can also be larger than the visible button. If you do not specify a hit frame, the objects in the Up state are used as the hit frame.

6. After you define the images of the four button states, choose Edit > Edit Movie to exit Symbol Edit mode.

7. Open the Library window by choosing Window > Library. Locate the button in the Library window and then drag the button symbol out of the Library onto the Stage. This step creates an instance of the button in the movie.

   For information on assigning actions to the button instance, see the documentation that applies the version of Flash that you are using. The documentation follows below:

**Assign a simple action to a button (Flash 5)**

1. In Edit Movie mode, select the button instance created in Step 7 above.

2. Choose Window > Actions to open the Actions panel.

3. In the Toolbox list on the left side of the panel, click the Basic Actions category to display the basic actions.

4. To assign an action, do one of the following:
   - Double-click an action in the Basics Actions category.
   - Drag an action from the Basic Actions category on the left to the Actions list on the right side of the panel.
   - Click the Add (+) button and choose an action from the pop-up menu.
   - Use the keyboard shortcut.

5. If the chosen action has any associated parameters, those parameters appear in the Parameter pane at the bottom of the Actions panel. (If the Parameter pane is not visible, click the small triangle in the lower right corner of the panel.) Choose or type the parameters appropriate for that action. For example, the `gotoAndPlay` action shown below contains three parameters: Scene, Type, Frame, and an option for Goto and Play.

**Assign a simple action to a button (Flash 4 and earlier)**

1. In Edit Movie mode, select the button instance created in Step 7 above.

2. Make sure that Enable Buttons from the Control menu is deselected.

3. Double-click the button to get the Instance Properties dialog box. **Note:** In Flash 2, this dialog box was the Link Properties: Button dialog box.
Assign the action by selecting the Action tab in the Instance Properties dialog box. Then, click the plus (+) button and double-click the appropriate action. **Note:** In Flash 2 assign the action using the Action pop-up menu in the Link Properties: Button dialog box. Only one action can be assigned to the button.

Make sure that Enable Buttons in the Control menu is checked, so that option is toggled back on.

If the selected action has any associated parameters, those parameters appear in the Parameter panel on the right side of the Actions panel. Choose or type the parameters appropriate for that action.

**Create advanced buttons**

After you've mastered a simple button, try more complex Flash buttons. You can create invisible buttons, buttons with animated states, and buttons with rollover effects.

Hit states are important to understand when designing a complex button. The button's shape and area represent the active area of the button. To experiment with the Hit state, do the following:

1. Introduce a button from the Common Libraries onto the Stage. Edit the button.
2. Highlight the frame in the button's Hit state, and insert a Keyframe. Test this behavior in a movie.
3. Resize the object in the Hit state dramatically, and test.
4. Delete the Keyframe in the Hit State and test the button.
5. Notice the effect on your button cursor, and the Up, Over and Down states that you see when the button is enabled.

**Invisible buttons**

The Up, Over and Down states of a button can be left empty. If these states are empty, it's necessary to define the Hit state so that it contains content.

When the Up state of the button is empty or invisible, the button is represented on the Stage by a blue shape. The shape is the equivalent to the content in the next Keyframe within the button. This blue shape isn't visible in your final movie. See the example below.

**Buttons with animated states**

To make animated buttons in Flash, place a Movie Clip in the button state that you are animating.

1. Create a movie clip for each state of the button that you want animated.
2. Create the button.
3. Place the movie clips in the button states to animate.
4. Place the button on the stage. **Note:** Movie clips cannot be tested in the Flash editor. Choose Control > Test Movie or export it as a SWF file to test it.

See the example below.

**A rollover in one area of the movie affects another area of the movie**

To create this effect, move the Hit state elements to a different area of the Stage than elements in the Over state of the button. This method works for simple rollover effects. The active area of the button is located in a different area onstage than the rollover effect. See the example below.

**Example**

**Note:** Flash Player is required to view the example.

- Download the Windows source file **advanced_buttons.zip** (17 K)
• Download the Mac OS source file advanced_buttons.sit (18 K)

Note: Flash 4 or later is required to open the source file.

Additional information
A good way to learn more about buttons and their construction is to study the buttons included as samples with Flash. In Flash 5, these sample buttons are available from the Windows > Common Libraries > Buttons menu. If you use Flash 4, you can access these buttons by choosing Libraries > Buttons. For Flash 2, the libraries of sample buttons are accessible from the Xtras menu.

A walkthrough of creating a button is also available in the Lessons that come with Flash 5. Choose Help > Lessons > 06 Buttons from within Flash to access this lesson.

Child display objects inaccessible after frame navigation
| ActionScript 3.0

Issue
When you navigate to a new frame in a movie clip in ActionScript 3.0 (for example, with the gotoAndStop() or gotoAndPlay() method), children of that clip aren’t accessible. The numChildren property correctly indicates the number of children that exist in the movie clip. However, attempting to access the children results in null values.

Solution
To have access to the children of the movie clip in the new frame, wait until they are instantiated and added to the display list. You can use two events, added (Event.ADDED) and render (Event.RENDER).

The ADDED event
In using the ADDED event, listen to the movie clip doing the navigation, or the parent movie clip. Each time a child of that clip in the new frame is instantiated and added to the display list, the added event is dispatched. Since the added event bubbles, this process happens for all children of that movie clip and all its children’s children, and so on. To check for children contained within your parent clip, you can test that the target of the ADDED event has a parent property equaling that clip.
// target movie clip contains one or more
// child movie clips on frame 2 that need
// to be accessed when the frame is reached
targetMovieClip.gotoAndStop(2);
// add a listener for the added event to
// indicate when a child is added to the target
targetMovieClip.addEventListener(Event.ADDED, targetChildAdded);
// called for each child as it is instantiated
// and added to the display list in targetMovieClip
function targetChildAdded(event:Event):void {
    // cast the event target as a DisplayObject
    var child:DisplayObject = DisplayObject(event.target);

    // if only accessing direct children of target
    if (child.parent == targetMovieClip) {
        // perform an action on this child
        // you can use a switch statement to
        // reference different actions on different
        // objects based on their name
        switch(child.name) {
            case "childA": {
                // childA actions
            }
            case "childB": {
                // childB actions
            }
            // etc.
        }
    }
}

Note: You still can’t reference a child object by its instance name within the ADDED event handler. That (automatic)
assignment doesn’t occur until after this event. Until then, the instance name variable can have a value of null unless it
existed in the previous frame.

The RENDER event
The RENDER event occurs right before Flash Player stage is drawn for that frame. All child movie clips are instantiated
and accessible. To dispatch the event, though, it’s necessary to call stage.invalidate(). Make this call from a movie clip
already within an active stage list or from a valid stage reference.
Additional information

The design of ActionScript 3.0 is different from ActionScript 2.0. ActionScript 3.0 doesn’t allow the same interaction between display objects on the timeline within Flash. When you use a goto command like gotoAndStop() or gotoAndPlay() in ActionScript 3.0, objects on the timeline within the new frame aren’t instantiated. Therefore, they’re not accessible and code on that frame can’t run until after the completion of the current script. Flash Player does, however, internally track the children about to be displayed in the new frame. Therefore, the display list is populated with null values. This process allows numChildren to display the correct and expected value even though valid references to the display objects on that frame don’t exist. If additional frame navigation occurs, the display list is again updated with the appropriate null children. However, the display objects within the previous frame would be saved from being instantiated. The frame ultimately rendered by Flash Player has changed.

Characters missing from text in Flash CS4 file opened in Flash CS5

Issue

When you open an Adobe Flash CS4 file with text in Flash CS5, characters are missing. For example, “You won the game” appears as “Yu wn h gam.”

Solution 1: Use Device Fonts.

Select the affected text blocks and set the anti-alias mode to Use Device Fonts. This setting uses the fonts that are on the end user’s computer. When the font used isn’t available, Flash uses a similar font in its place.

Solution 2: Embed the fonts.

In Flash CS5, you manage font embedding in the Font Embedding dialog box.

1 Choose Text > Font Embedding. You can also open it via the Text Property Inspector and the Properties of a Font Symbol in the Library.

Last updated 11/6/2015
Add the fonts and character ranges you need for your text fields and click OK. When you create the SWF file, Flash embeds all of the characters necessary to render the text field. The text always looks as it was designed to, even if the font isn’t available on an end user’s computer. However, font embedding can make your SWF file larger.

For more information on Font Embedding and Anti-Alias modes, see Embed fonts for consistent appearance in the Flash online Help.

Additional information

Missing characters occur in files that have the following characteristics:

- The file contains ActionScript code that programmatically changes the text of a text field at runtime.
- The file has selected an anti-alias mode other than Use Device Fonts.
- The file is not embedding the necessary fonts and character ranges.

If the anti-alias mode in Flash CS4 isn’t Use Device Fonts, and there’s no embedded text, Flash uses device fonts when creating SWF files. Font embedding is on a per-text-field basis, so it is easy to determine whether a font is embedded. Flash CS4 uses Device Fonts automatically, without the user knowing it, so the SWF file works correctly on the local authoring computer. The problem isn’t noticeable until the SWF file is published and tested on an end user’s computer.

In Flash Pro CS5, font embedding is document-based, rather than on a field-by-field basis as it is in Flash CS4. Flash CS5 automatically embeds characters in text fields on the Stage if an anti-alias mode other than Use Device Fonts is selected. So, it’s unnecessary to revert to the Use Device fonts setting.

This behavior creates a problem for older files with ActionScript. If a Classic text field has embedded fonts, but no fonts are embedded in the SWF file, the Classic text field is blank or missing characters.

Can't install Android app | Flash CS5 CS5.5

Issue

When you publish an AIR for Android application in Flash Professional CS5 or CS5.5, the application doesn’t install on the connected device. There is insufficient available disk space.

Note: This issue also occurs when using the AIR Developer Tool (ADT) command line -installApp command.

Solution

Check the size of the .apk (AIR for Android package format) file created and compare it to the amount of available disk space on your device. If the size of the .apk file is greater than the available space on the device, free up disk space before the application can be installed.
Can't install an AIR for Android app to offline device | Flash CS5, CS5.5

**Issue**
When you publish an AIR for Android application in Flash Professional CS5 or CS5.5 to a device that is offline, the installation never finishes. The progress bar never goes away. (An offline device is one that the computer does not recognize as plugged in.)

**Solution**
Follow the steps below on the connected device:
1. Unplug the device from the USB cable and plug it back in.
2. Turn USB Debugging OFF and then ON again by following these steps.
   a. Select the Home button on the device.
   b. Press the Options button on the device and select the Settings option.
   c. Navigate to the Applications > Development option and find the USB Debugging preference.
   d. Turn USB debugging OFF and then ON again.

Cannot type GB18030 characters in panels and dialog boxes

**Issue**
When running the Simplified Chinese version of Flash Professional on Windows XP, you cannot use GB18030 characters in user interface panels and dialog boxes.

**Solutions**

**Solution 1: Upgrade to Windows Vista/Windows 7.**
If you replace Windows XP with Windows Vista/Windows 7, you can run Flash Professional using the GB18030 character set. It's unnecessary to install the Flash GB18030 workaround.

**Solution 2: Install the Flash CS4/CS5/CS5.5/CS6 GB18030 workaround.**
To install the workaround, do the following:
1. Install the Microsoft GB18030 Support Package from
2 Close Flash Professional.
3 Run GB18030_Enable.bat (the Flash GB18030 workaround included in the download below).

FlashCS4_GB18030_workaround.zip
FlashCS5_GB18030_workaround.zip
FlashCS5_5_GB18030_workaround.zip
FlashCS6_GB18030_workaround.zip

4 Start Flash Professional.

To uninstall, do the following:
1 Run GB18030_Disable.bat (also part of the Flash GB18030 workaround included in the download mentioned above).

2 Flash Professional now runs as normal without the GB18030 workaround. The benefit to uninstalling is that it allows users more flexibility when using non-GB18030 characters in the user interface.

**Description of the Flash Professional CS6, CS5.5, CS5 or CS4 GB18030 workaround**

The workaround modifies all of the persistent user interface that modeless panels represent. The workaround uses a registry switch to enable modeless panels to accept GB18030 characters. To disable the use of GB18030 characters (returning Flash Professional to its default mode), follow the instructions above for uninstalling the workaround. It is based on a Microsoft GB18030 font that is already on a GB18030 system. (it is also installed when a user installs the Microsoft GB18030 Support Package referenced above.)

The workaround also modifies a few of the most important modal dialogs so that they can accept GB18030 characters. Not all dialog boxes have been modified, so even with the workaround enabled, many dialogs don’t properly accept GB18030 characters.

**Additional information**

GB18030 is the registered name for the official character set of the People’s Republic of China (PRC). It is the character set that users of the Simplified Chinese version of Flash Professional can use when typing into the Flash Professional user interface. There is a bug in the Windows XP operating system that prevents the GB18030 character set from working correctly. The purpose of the Flash workaround described below is to address this issue.

### Calling Flash ActionScript functions using Lingo

**Flash files in Director MX 2004**

Director MX 2004’s object model allows direct access to an SWF movie clip and ActionScript functions. ActionScript functions can be called directly without setting up an object reference using `getVariable()`. Movie clips can be referenced via Lingo or JavaScript syntax.

```lingo
--example 1 global myObj on callFlashFunction me myObj=myObj=myClip=myClip.Calculate() end
```

---

**Setting up the object reference and calling a function**

1. The following custom handler first returns an object reference to a movie clip named ‘myClip’. The second statement fires off a function named ‘Calculate()’ in the same movie clip.

   ```lingo
   --example 1 global myObj on callFlashFunction me myObj=sprite(flashSprite).myClip.myObj.Calculate() end
   ```

---

---
To use the newObject command in Director, use either the Flash sprite command or the global Flash command. For more information see "newObject()" in the Director Lingo Dictionary. There is also some downloadable sample code in Global Flash command for creating an ActionScript object (TechNote 16691).

**Flash files in Director MX**

When using Director MX, object references can be made to a Flash movie timeline. This functionality allows control over Flash movies and individual Flash objects. An object reference is created by calling a getVariable() command on a specific sprite. The getVariable() command is used in two ways: to return the literal value of a Flash variable (TRUE) or to return an object reference (FALSE). For more information see "getVariable()" in the Director MX Lingo Dictionary. Also, see the following section in Director MX Help: Media > Using Flash and Other Interactive Media Types. Using getVariable() and object references to control a Flash sprite The code below demonstrates how to use getVariable() and object references to control a Flash sprite.

### Setting up the object reference

Set up the object reference to the main timeline of the Flash movie using the getVariable() command. This is achieved by setting the "returnValueOrReference" parameter to FALSE. The following script creates a reference to the timeline.

```
global myObj on beginSprite me myObj=getVariable(sprite(flashSprite),"_level0",False) end
```

### Using the object reference

Once the reference has been set up, Lingo can be used to call user-defined functions in the Flash movie. For example, the following script will call a custom Flash function 'doSomething()' in the timeline of the main Flash movie:

```
global myObj myObj.doSomething()
```

The object reference will handle parameters if the function has been defined to accept parameter information.

The reference can also be used to control the timeline of the Flash movie using standard ActionScript. This statement sends the play head of the Flash movie to frame 10 and instructs the movie to play from that frame:

```
global myObj myObj.goToAndPlay(10)
```

An object reference can be expanded to control the play head of a movie clip and call user-defined functions that are defined in the timeline of the movie clip. The first statement of this sample script sends the play head of a movie clip named 'myClip' to frame 20. The second statement fires off a function named Calculate():

```
global myObj myObj.myClip.goToAndStop(20) myObj.myClip.Calculate()
```

### Tips and Tricks

Last updated 11/6/2015
### Bezier Pen Tool | Flash Professional CS5

**Introduction**

Flash CS5 introduced a Bezier Pen Tool and a Subselection Tool for PostScript style curve editing. Points on Bezier curves change after certain operations drawing or editing operations. This effect is natural and doesn’t affect the final size of the file. The following article describes the technical details behind these curve operations.

**Quadratic Beziers and Cubic Beziers**

The SWF file format describes curves as Quadratic Bezier curves. These curves are a little different from the Cubic Beziers that PostScript uses. As the SWF file format documentation describes: “SWF uses Quadratic Bezier curves because they can be stored more compactly, and can be rendered more efficiently.”

Quadratic Beziers have only one control handle for each segment of the curve. In a PostScript drawing tool, two nodes on the curve, and two control handles off the curve define each curve segment. There are a total of four data points. The SWF file format also describes the two on-the-curve nodes. However, it’s only necessary to describe one off-the-curve control handle. Therefore, it can describe the same curve with three data points.

Flash 5 introduced a Bezier Pen Tool and a Subselection Tool. This tool can create and edit alternate Cubic Bezier representations of all of the native Quadratic Bezier curves. The core representation remains the native Quadratic Bezier curve. However, Cubic Bezier curves can be generated on demand to enable Bezier Pen drawing and PostScript style editing.

This Cubic Bezier representation is stored in the FLA file for use in subsequent edits. During some editing operations, a new Cubic Bezier representation is generated from the native Quadratic Bezier representation. When the FLA is compiled to SWF file, only the compact Quadratic Bezier is included.

---

<table>
<thead>
<tr>
<th></th>
<th>Use the new Object Inspector to examine the hierarchy of elements inside Flash MX sprites. The ability to quickly examine all your movie components reduces both debugging and development time. For example enter sprite reference into the OI and explore the results: sprite(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When working in Flash, make sure to assign unique instance names to movie clips. This will make it easier to reference these objects in Director.</td>
</tr>
<tr>
<td></td>
<td>When working with a specific movieclip that is deeply nested within other movieclips, it is advisable to create an object reference that will directly point to that specific movieclip.</td>
</tr>
<tr>
<td></td>
<td>Flash movies that are loaded using theloadMovie() ActionScript command can also be targeted. To do this, pass the specific level of the movie using the parameter variableName within thegetVariable() command.</td>
</tr>
</tbody>
</table>

**Additional Information**

- How do I use `getVariable` and `setVariable` with a Flash movie? (TechNote 14798)
- How do I pass a Variable from Flash to Director? (TechNote 14796)
- Using Flash cast members in Director movies
- Global Flash command for creating an ActionScript object (TechNote 16691)
To summarize, PostScript style control handles require a different type of curve than that which Flash Player renders. The Flash 5 authoring application can automatically generate these four-point curves. It tries to reuse them when possible, but the underlying description is still a three-point Quadratic Bezier curve.

**What causes curve regeneration?**

The Flash authoring application combines painterly and vector drawing techniques, and you can do things that are impossible in other vector programs. Some of these operations aren’t handled well by PostScript style curves. In such a case, the application creates a new Cubic Bezier curve. The next time you use a Bezier Pen, you could see that the points have changed.

For example, if you import some vector art in the Adobe Illustrator AI format, it starts with a Cubic Bezier representation. It’s then converted to Flash’s internal Quadratic Bezier curves. The Cubic Bezier points remain for later PostScript style editing. But if you use Flash’s Lasso Tool to select part of this curve and delete it, then an entirely new Cubic Bezier curve is created. It’s easy to grab and delete parts of curves in Flash. However, this nonPostScript style of vector drawing requires a new PostScript style curve.

Another example is Flash’s Vector Eraser. If you draw a circle and select it with the Subselection Tool, you can see that the shape has few Bezier control handles. If you then swipe the Eraser across this vector shape, then a new PostScript representation is created. The new Bezier Pen points are different from the old points.

The same holds true for Flash’s Vector Brushes, or the ability to directly grab and twist and sculpt the edges of a shape, or the way that Flash has built-in Boolean operations on shapes. All of these nonPostScript drawing methods can force Flash to render a new PostScript style Cubic Bezier representation.

There are additional subtle regeneration cases. If you are drawing with the Bezier Pen Tool or editing with the Subselection Tool, then performing an Undo command can often cause regeneration. In this case, you can see more points than were originally there. Many simpler curves sometimes represent a complex curve. Moving a shape with the regular Arrow Tool can also give the appearance of many more Bezier Pen Tool control points. This effect is because of the way the transform is currently handled.

**Do extra Bezier Pen Tool points matter?**

For total file size, it doesn’t matter how many four-point Cubic Bezier points there are in a curve. The Cubic Bezier representation is not included in the final SWF file. All that matters is the overall complexity of the regular Quadratic Bezier curve. All that matters is the overall smooth appearance of the curve.

You can confirm this statement for yourself by making a symbol containing a simple Cubic Bezier curve using the Bezier Pen Tool. Duplicate the symbol, and move some of the points with the Subselection Tool and Undo until you see the multiple simpler curves. These two symbols describe the same overall curve, but one uses more four-point Cubic Bezier curves to do so. When you Publish to SWF file and inspect the size report, you see that the final symbols are about the same size.

All that matters in the final SWF file is the number of underlying three-point Quadratic Bezier curves. The extra nodes and handles seen with the Subselection Tool and the Bezier Pen Tool are not included in the final file.

If you’re importing a shape from a PostScript file then there can be a different problem. The original shape could be overly complex, and the original Quadratic Bezier conversion could have more points than required. Whenever you’re bringing in graphics from a print-based application, it can be useful to examine each curve and use Flash’s Smoothing and Optimize commands. These commands can help in reducing the overall file size.
**Drawing techniques**

In all cases, examine the natural-drawing techniques in Flash itself. These techniques don’t use a four-point Pen metaphor, but the natural brushing, erasing, and combining operations can give optimized results quickly.

**Summary**

The native curve in a SWF file is a Quadratic Bezier, with two on-the-curve points and one off-the-curve control handle. When you use the Bezier Pen Tool or Subselection Tool in Flash, then the application automatically creates a PostScript style Cubic Bezier. The curve has two on-the-curve points and two off-the-curve control handles. This Cubic representation is only used in editing, and the points can change during the editing. Only the compact Quadratic Bezier is compiled into the final SWF file.

**Adding actions to shared buttons**

Shared libraries in Macromedia Flash allow multiple movies to reference a single media asset such as a graphic, movie clip or sound. During authoring, shared symbols can improve workflow by allowing teams of designers to easily share media. During playback, shared assets download once and can be used by several movies, reducing file size and download time.

Due to the nature of shared symbols, however, actions cannot be applied to instances of shared buttons. One solution is to place an instance of the button inside an empty movie clip and apply actions to that button instance. This "parent" movie clip symbol is then shared, rather than the button itself. However, this results in the same action being executed for each instance of that shared movie clip. Another solution is to use shared graphics inside a locally created button symbol. Different actions could be attached to separate instances of this button symbol, since it is local to the current movie. This method, however, requires keeping track of multiple imported graphic symbols.

Smart clips, new in Flash 5, provide one way to easily allow different instances of the same shared button symbol to perform different actions. Similar to the first solution mentioned above, a button symbol is "nested" inside another movie clip. Actions applied to the button include ActionScript variables, such as parameters to functions or actions. Clip parameters that correspond to those ActionScript variables are added to the parent movie clip, which is then exported as a shared item. In movies that import this movie clip, it’s easy to customize the behavior of each button/clip instance using the Clip Parameters panel.

**An example** The steps below explain how to create a shared button whose instances will open different URLs from a getURL() action. The first part involves creating the shared button and assigning clip parameters to its parent clip.

**To make the shared button symbol:**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Open a new movie in Flash and save it as shared.fla.</td>
</tr>
<tr>
<td>2</td>
<td>Create a new movie clip symbol (Insert &gt; Symbol) and give the symbol a name like button_parent, for example.</td>
</tr>
<tr>
<td>3</td>
<td>Add a button to this movie clip's Timeline, such as one of the buttons from the Buttons common library (Window &gt; Common Libraries &gt; Buttons).</td>
</tr>
</tbody>
</table>
Attach the following action to the button instance:

```
{ getURL (theURL); }
```

If working in Normal scripting mode, be sure to select the Expression option next to the URL field.

Open the Library window (Window > Library) and select the button_parent symbol. Choose Define Clip Parameters from the Library window’s Options menu.

Click the Add (+) button at the top of the Clip Parameters dialog to add a new name/value parameter pair.

Select the Name field by double-clicking it, and enter theURL as the parameter name, as shown below.

Leave the other fields at their default values.

Click OK.

Select the button_parent symbol in the Library window again, and choose Linkage from the Options menu.
The next step is to import button_parent into a new, host movie and add instances of the clip to the Timeline. The Clip Parameters panel is used to modify the getURL action applied to each button instance.

To import the movie clip and customize the getURL action:

1. Open a new Flash movie and save it as host.fla to the same folder that contains shared.fla and shared.swf.
2. Choose File > Open as Shared Library and open shared.fla. The Library window for that FLA should appear containing the button_parent symbol as well as the actual button (named Grey_button, in this case).
3. Drag two or three instances of button_parent (not the button symbol itself) from shared.fla's the Library window to the stage. (See note below about nested shared symbols.)
4. Select one of the movie clip instances and open the Clip Parameters dialog (Window > Panels > Clip Parameters).
### A note about "nested" shared symbols

When you import a shared symbol that contains other symbols, those "nested" symbols also appear in the host movie's Library window. This is the case even if the nested symbols are not shared themselves. For instance, in the above example, the movie clip named button_parent contains a button named Grey_button. Just the movie clip has been shared and imported, but both the button and its parent movie clip appear in the host movie's Library.

If you attempt to edit the shared movie clip symbol, button_parent, Flash presents you with the following warning:

Shared symbols can only be edited, and remained linked, if changes are made to the symbol in its original movie (shared.fla, in this case.) If you click Yes to the above warning dialog, the shared symbol will be fully imported into the host movie. That is, it will no longer be linked.

However, the same warning does not appear if you attempt to edit Grey_button, a symbol nested inside of the shared symbol button_parent. This is somewhat misleading as nested symbols appear only as placeholders for their linked counterparts. For example, changes made to Grey_button will only appear while authoring the host movie in Flash. During playback, the external Library item is referenced and displayed.

**Changing the appearance of the imported button** To change the appearance of the imported button symbol, open shared.fla and make the desired changes to the original button symbol. Re-export the Flash movie as an SWF to its same location.