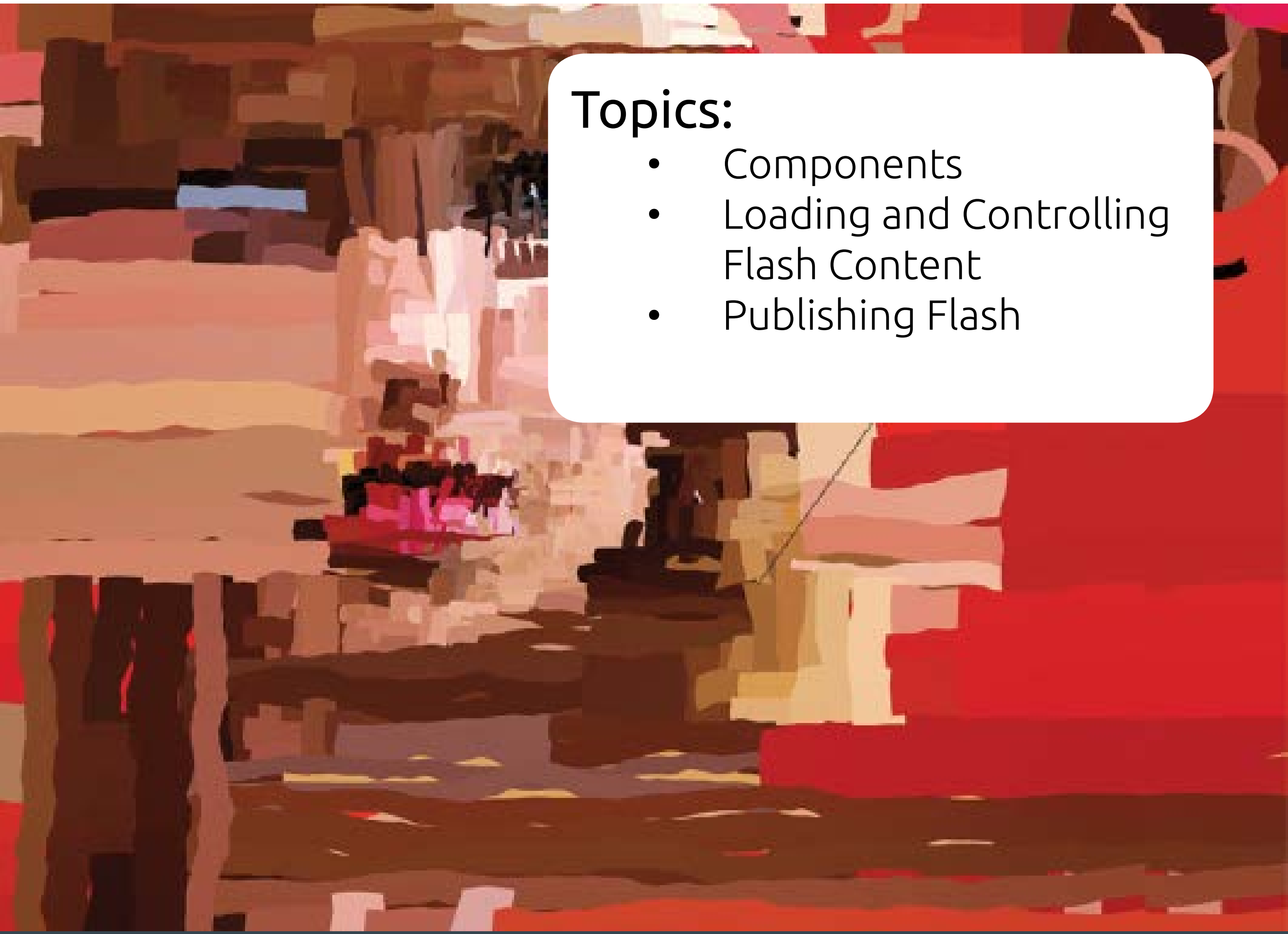
Adobe Flash CS 6

Level 4



Topics:

- Components
- Loading and Controlling Flash Content
- Publishing Flash

Components

Adobe Flash Professional components are building blocks for creating rich interactive applications on the web. By providing complex controls that behave in a consistent way and are ready to use and customize, components significantly reduce the time and effort needed to develop applications from scratch.

For example, rather than creating your own custom button, combo box, list, or video player, you can simply drag a component from the Components panel into your document during authoring. You can easily customize the look and feel of these components to suit your application design. Components share runtime libraries; once you've used one component in your application, adding another one does not greatly impact the size of your SWF file at runtime. Explore the sections below to dive into whatever topic interests you.

Each component includes an ActionScript application programming interface (API) that allows you to customize it at runtime. The API allows you to handle events that occur when a user interacts with a component or when something significant happens to it. The API also enables you to set properties and call methods and to apply styles and skins to customize a component's appearance. Using ActionScript, you can also create a component instance at runtime and configure it as needed.

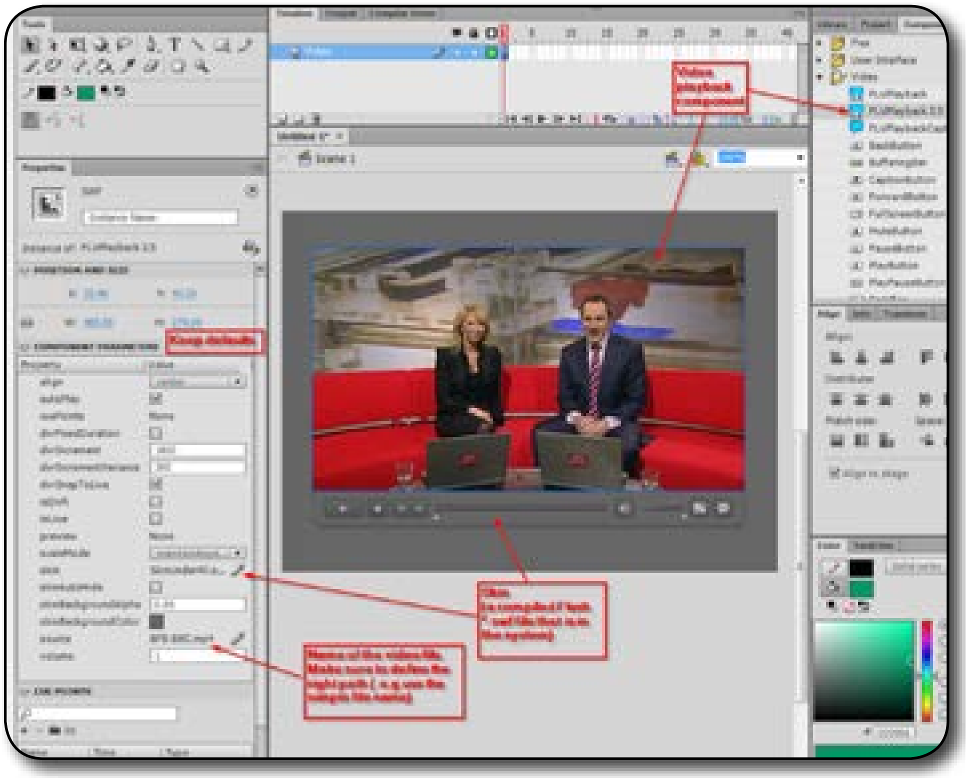
Adobe Flash Professional includes both ActionScript 2 and ActionScript 3 components. You cannot mix these two sets. When you create a new Flash document, Flash Professional presents either ActionScript 3 components or ActionScript 2 components based on whether you choose an ActionScript 3 FLA file or an ActionScript 2 FLA file. When you open an existing FLA file, Flash determines which set of components to present based on whether the publish settings specify ActionScript 2 or ActionScript 3.

Using the Flash video component

Using Flash CS6 (and CS5) component is really easy

- You can use *.mp4 videos without prior encoding to a Flash video format
- All operations are centralized in the properties panel
- The only difficulties relate to file path operations (correct paths and copying files)

In most cases, you would have to adapt your video file, i.e. at least make it smaller and cut of unwanted beginning and end.
Just below we explain the procedure for an *.mp4, *.flv or *.f4v video that already is "ready" for production.


Video playback component
Keep defaults
Skin (a compiled Flash *.swf file that is in the system)
Name of the video file. Make sure to define the right path (e.g. use the simple file name)

Step 1: Drag the component to the stage and save the file

1. Open the components panel (***Menu Window->Components*** or ***CTRL-F7***)
2. Drag FLPLayback 2.5 on the stage
3. Now save your Flash file, else it will not work, since Flash will not find the video file.
4. Copy the video file into the same directory as your flash file (unless you know how to deal with relative file paths, something that you may have learned creating HTML pages using pictures)

Step 2: Configure the properties of the component

1. Select the component
2. Set the name of the video file with source in the Properties panel (e.g. BFB-BBC.mp4). Make sure to shorten the file path
    - Bad: C:\...\flash\ex6\screenshots\my_video.mp4
    - Good: my_video.mp4
3. Select an appropriate skin. A skin will define what kinds of controls the user will have. You also can make adjustements to the color
4. Keep the defaults for starters, e.g. maintainAspectRation for starters.

Step 3: Publishing a flash file that uses video
If you plan to publish the flash file on a web site or if you mail your application, do not forget to include all the files, for example

- flash-cs6-mp4-video.html (the HTML file, optional)
- flash-cs6-mp4-video.swf (the Flash file)
- BFB-BBC.mp4 (the video)
- SkinUnderAllNoFullscreen.swf (the skin library)

The Loader class is used to load SWF files or image (JPG, PNG, or GIF) files. Use the load() method to initiate loading. The loaded display object is added as a child of the Loader object.
Use the URLLoader class to load text or binary data.

The Loader class overrides the following methods that it inherits, because a Loader object can only have one child display object—the display object that it loads. Calling the following methods throws an exception: addChild(), addChildAt(), removeChild(), removeChildAt(), and setChildIndex(). To remove a loaded display object, you must remove the Loader object from its parent DisplayObjectContainer child array.

iOS notes

```
 var loader:Loader = new Loader();
 var url:URLRequest = new URLRequest("swfs/SecondarySwf.swf");
 var loaderContext:LoaderContext = new LoaderContext
(false, ApplicationDomain.currentDomain, null);
 loader.load(url, loaderContext);
```

In AIR applications on iOS, you can only load a SWF file containing ActionScript from the application package. This restriction includes any ActionScript, such as assets with class names exported for ActionScript. For loading any SWF file, you must load the SWF using the same application domain as the parent SWF, as shown in the following example:

In addition, on iOS you can't load a SWF file that contains any ActionScript ByteCode (ABC) then unload it and reload it. If you attempt to do this, the runtime throws error 3764.

Prior to AIR 3.6, only SWF files that do not contain ActionScript bytecode can be loaded, regardless of whether they're loaded from the application package or over a network. As an alternative to using an external SWF file with ActionScript, create a SWC library and link it in to your main SWF.

AIR 3.7 and higher supports loading of externally hosted secondary SWFs. The detailed description about this feature can be found here.

These iOS restrictions restrictions do not apply when an application is running in the iOS Simulator (ipa-test-interpreter-simulator or ipa-debug-interpreter-simulator) or interpreter mode (ipa-test-interpreter or ipa-debug-interpreter.)

Loader security

When you use the Loader class, consider the Flash Player and Adobe AIR security model:

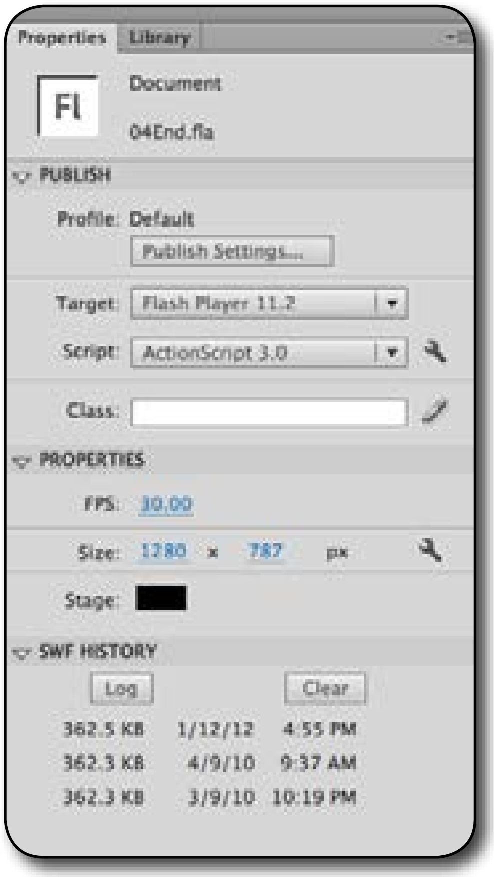
You can load content from any accessible source.

- Loading is not allowed if the calling SWF file is in a network sandbox and the file to be loaded is local.
- If the loaded content is a SWF file written with ActionScript 3.0, it cannot be cross-scripted by a SWF file in another security sandbox unless that cross-scripting arrangement was approved through a call to the System.allowDomain() or the System.allowInsecureDomain() method in the loaded content file.
- If the loaded content is an AVM1 SWF file (written using ActionScript 1.0 or 2.0), it cannot be cross-scripted by an AVM2 SWF file (written using ActionScript 3.0). However, you can communicate between the two SWF files by using the LocalConnection class.
- If the loaded content is an image, its data cannot be accessed by a SWF file outside of the security sandbox, unless the domain of that SWF file was included in a URL policy file at the origin domain of the image.
- Movie clips in the local-with-file-system sandbox cannot script movie clips in the local-with-networking sandbox, and the reverse is also prevented.
- You cannot connect to commonly reserved ports. For a complete list of blocked ports, see “Restricting Networking APIs” in the ActionScript 3.0 Developer’s Guide.
- However, in AIR, content in the application security sandbox (content installed with the AIR application) are not restricted by these security limitations.

When loading a SWF file from an untrusted source (such as a domain other than that of the Loader object’s root SWF file), you may want to define a mask for the Loader object, to prevent the loaded content (which is a child of the Loader object) from drawing to portions of the Stage outside of that mask, as shown in the following code:

```
import flash.display.*;
import flash.net.URLRequest;
var rect:Shape = new Shape();
rect.graphics.beginFill(0xFFFFFF);
rect.graphics.drawRect(0, 0, 100, 100);
rect.graphics.endFill();
addChild(rect);
var ldr:Loader = new Loader();
ldr.mask = rect;
var url:String = “http://www.unknown.
example.com/content.swf”;
var urlReq:URLRequest = new URLRe-
quest(url);
ldr.load(urlReq);
addChild(ldr);
```

Publishing overview



Document
04End.fla
Profile: Default
Target: Flash Player 11.2
Script: ActionScript 3.0
Class:
FPS: 30.00
Size: 1280 x 787 px
Stage:

362.5 KB	1/12/12	4:55 PM
362.3 KB	4/9/10	9:37 AM
362.3 KB	3/9/10	10:19 PM

Document Property inspector

Note:
In Flash Professional CS5, when you set the Flash Player target to Flash Player 10 in the Publish Settings, the target is actually Flash Player 10.1.

You can play content in the following ways:

- In Internet browsers that are equipped with Flash Player
- As a stand-alone application called a projector
- With the Flash ActiveX control in Microsoft Office and other ActiveX hosts
- With Flash Xtra in Director® and Authorware® from Adobe®

By default, the Publish command creates a Flash Pro SWF file and an HTML document that inserts your Flash Pro content in a browser window. The Publish command also creates and copies detection files for Macromedia Flash 4 from Adobe and later. If you change publish settings, Flash Pro saves the changes with the document. After you create a publish profile, export it to use in other documents or for others working on the same project to use.

When you use the Publish, Test Movie, or Debug Movie commands, Flash creates a SWF file from your FLA file. You can view the sizes of all the SWF files created from the current FLA file in the Document Property inspector.

Flash® Player 6 and later support Unicode text encoding. With Unicode support, users can view multilanguage text, regardless of the language that the operating system running the player uses.

You can publish the FLA file in alternative file formats—GIF, JPEG, and PNG —with the HTML needed to display them in the browser window. Alternative formats allow a browser to show your SWF file animation and interactivity for users who don't have the targeted Adobe Flash Player installed. When you publish a Flash Pro document (FLA file) in alternative file formats, the settings for each file format are stored with the FLA file.

You can export the FLA file in several formats, similar to publishing FLA files in alternative file formats, except that the settings for each file format are not stored with the FLA file.
Alternatively, create a custom HTML document with any HTML editor and include the tags required to display a SWF file.
To test how the SWF file works before you publish your SWF file, use Test Movie (**Control > Test Movie > Test**) and Test Scene (**Control > Test Scene**).

HTML documents

You need an HTML document to play a SWF file in a web browser and specify browser settings. To display a SWF file in a web browser, an HTML document must use the object and embed tags with the proper parameters.

Flash Pro can create the HTML document automatically when you publish a SWF file.

Note:
You can generate an HTML document using the correct object and embed tags using the Publish Settings dialog box, and selecting the HTML option. For more information, see Specify publish settings for HTML documents .

Detecting whether Flash Player is present

In order for your published Flash Pro content to be seen by Web users, Flash Player must be installed in their Web browser.

The following resources and articles provide up-to-date information about how to add code to your web pages to determine if Flash Player is installed and provide alternative content in the page if it is not.

- Flash Player Help
- Flash Player Detection Kit
- Adobe Flash Player version checking protocol
- Future-Proofing Flash Player Detection Scripts

Publishing for mobile devices

Adobe® AIR® for Android® and iOS® lets Flash Pro users create engaging content for mobile devices using the ActionScript® scripting language, drawing tools, and templates. For detailed information on authoring for mobile devices, see the AIR Developer Reference and the Content Development Kits in the Mobile and Devices Development Center.

Note:
Depending on the mobile device for which you are developing, certain restrictions can apply as to which ActionScript commands and sound formats are supported. For more details, see Mobile Articles on the Mobile and Devices Development Center.

Testing mobile content with the Mobile Content Simulator

Flash Pro also includes a Mobile Content Simulator, a way to test content created with Adobe AIR in an emulated Android or iOS environment. With the Mobile Content Simulator, you can use the Control > Test Movie command to test your Flash file in the AIR Debug Launcher for Mobile, which in turn launches the simulator.

Once the simulator window is opened, you can send input to the Flash file as if it were running on a mobile device. The inputs
available include:

- Accelerometer, X, Y, and Z axes
- Orientation threshold angle
- Touch and gestures, including pressure sensitivity
- Geolocation, direction, and velocity
- Hardware keys (found on Android devices)

Publishing secure Flash documents

Flash Player 8 and later contain the following features that help you ensure the security of your Flash Pro documents:

Buffer overrun protection
Enabled automatically, this feature prevents the intentional misuse of external files in a Flash Pro document to overwrite a user's memory or insert destructive code such as a virus. This prevents a document from reading or writing data outside the document's designated memory space on a user's system.

Exact domain matching for sharing data between Flash documents
Flash Player 7 and later enforce a stricter security model than earlier versions. The security model changed in two primary ways between Flash Player 6 and Flash Player 7:

Exact domain matching
Flash Player 6 lets SWF files from similar domains (for example, *www.adobe.com and store.adobe.com*) communicate freely with each other and with other documents. In Flash Player 7, the domain of the data to be accessed must match the data provider's domain exactly for the domains to communicate.

HTTPS/HTTP restriction
A SWF file that loads by using nonsecure (non-HTTPS) protocols cannot access content loaded by using a secure (HTTPS) protocol, even when both protocols are in exactly the same domain.

Local and network playback security
Flash Player 8 and later include a security model that lets you determine the local and network playback security for SWF files that you publish. By default, SWF files are granted read access to local files and networks. However, a SWF file with local access cannot communicate with the network, and the SWF file cannot send files or information to any networks.

Allow SWF files to access network resources, letting the SWF file send and receive data. If you grant the SWF file access to network resources, local access is disabled, protecting information on the local computer from potentially being uploaded to the network.

To select the local or network playback security model for your published SWF files, use the Publish Settings dialog box.

Flash Player

Flash Player plays Flash Pro content in the same way as it appears in a web browser or an ActiveX host application. Flash Pro Player is installed with the Flash Pro application. When you double-click Flash Pro content, the operating system starts Flash Player, which then plays the SWF file. Use the player to make Flash Pro content viewable for users who aren't using a web browser or an ActiveX host application.
To control Flash Pro content in Flash Player, use menu commands and the fscommand() function.

Use the Flash Player context menu to print Flash Pro content frames.

Do one of the following:

- To open a new or existing file, select **File > New,** or **Open**.
- To change your view of the application, select **View >Magnification** and make a selection.
- To control Flash Pro content playback, select **Control > Play, Rewind, or Loop Playback**.

Wrap-up

By the end of this workshop, you should be able to:

- Understand what Components are.
- Able to use the Loader Class in Flash
- Publishing FLash files