MX

macromedia®

FLASH™MX 2004

Using Flash

Acknowledgments

Director: Erick Vera

Project Management: Julee Burdekin, Erick Vera

Writing: Jay Armstrong, Jody Bleyle, Mary Burger, Francis Cheng, Jen deHaan, Stephanie Gowin, Phillip Heinz, Shimul Rahim, Samuel R. Neff

Managing Editor: Rosana Francescato

Editing: Mary Ferguson, Mary Kraemer, Noreen Maher, Antonio Padial, Lisa Stanziano, Anne Szabla

Production Management: Patrice O'Neill

Media Design and Production: Adam Barnett, Christopher Basmajian, Aaron Begley, John Francis

Second Edition: June 2004

Macromedia, Inc.
600 Townsend St.
San Francisco, CA 94103

CONTENTS

CHAPTER 1
Working with Flash Documents

When you create and save Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 documents within the Flash authoring environment, the documents are in FLA file format. To display a document in Macromedia Flash Player, you must publish or export the document as a SWF file.

Note: For information on publishing or exporting a file, see Chapter 15, "Publishing," on page 309 or Chapter 16, "Exporting," on page 345.

You can add media assets to a Flash document and manage the assets in the library, and you can use the Movie Explorer to view and organize all the elements in a Flash document. The Undo and Redo commands, the History panel, and the Commands menu let you automate tasks in a document.

This chapter contains the following sections:

Creating or opening a document and setting properties

You can create a new document or open a previously saved document as you work in Flash. In Windows, you can use the New File button to open a document of the same type as the last document created.

To set the size, frame rate, background color, and other properties of a new or existing document, you use the Document Properties dialog box. You can also use the Property inspector to set properties for an existing document. The Property inspector makes it easy to access and change the most commonly used attributes of a document. For more information on the Property inspector, see "Using panels and the Property inspector" in *Getting Started with Flash*.

You can open a Flash template as a new document. You can select from standard templates that come with Flash or open a template you have already saved. For information on saving a document file as a template, see "Saving Flash documents" on page 14.

In the On Launch section of the Preferences dialog box, you can select an option to specify what document Flash opens when you start the application: You select New Document to open a new, blank document, Last Documents Open to open the documents that were open when you last quit Flash, or No Document to start Flash without opening a document. For more information, see "Setting preferences in Flash" in *Getting Started with Flash*.

For information on creating a new document using the Start page, see "Using the Start page" in *Getting Started with Flash*.

You can open a new window as you work.

To create a new document:

1. Select File > New.
2. On the General tab, select Flash Document.

To create a new document with the New File button (Windows only):

- Click the New File button in the main toolbar to create a new document of the same type as the last document created.

To open an existing document:

1. Select File > Open.
2. In the Open dialog box, navigate to the file or enter the path to the file in the Go To text box.
3. Click Open.

To set properties for a new or existing document in the Document Properties dialog box:

1. With the document open, select Modify > Document.

   The Document Properties dialog box appears.
2. For Frame Rate, enter the number of animation frames to appear every second. For most computer-displayed animations, especially those playing from a website, 8 frames per second (fps) to 12 fps is sufficient (12 fps is the default frame rate).
3. For Dimensions, do one of the following:
   - To specify the Stage size in pixels, enter values in the Width and Height text boxes.

     The default document size is 550 x 400 pixels. The minimum size is 1 x 1 pixels; the maximum is 2880 x 2880 pixels.
   - To set the Stage size so that there is equal space around the content on all sides, click the Contents button to the right of Match. To minimize document size, align all elements to the upper left corner of the Stage, and then click Contents.
   - To set the Stage size to the maximum available print area, click Printer. This area is determined by the paper size minus the current margin selected in the Margins area of the Page Setup dialog box (Windows) or the Print Margins dialog box (Macintosh).
   - To set the Stage size to the default size, click Default.
4. To set the background color of your document, click the triangle in the Background Color box and select a color from the palette.
5. To specify the unit of measure for rulers that you can display along the top and side of the application window, select an option from the pop-up menu in the upper right. For more information, see "Using the grid, guides, and rulers" in *Getting Started with Flash*.(This setting also determines the units used in the Info panel.)
6. Do one of the following:
   - To make the new settings the default properties for your new document only, click OK.
   - To make the new settings the default properties for all new documents, click Make Default.

To create a new document from a template:

1. Select File > New.
2. Click the Templates tab.
3. Select a category from the Category list, and select a document from the Category Items list.
4. Click OK.

To open a new window in the current document:

- Select Window > New Window.

To change document properties with the Property inspector:

1. Deselect all assets, then select the Selection tool.
2. If the Property inspector is not visible, select Window > Properties.
3. Click the Size control to display the Document Properties dialog box and access its settings.
4. To select a background color, click the triangle in the Background color box and select a color from the palette.
5. For Frame Rate, enter the number of animation frames to appear every second.
6. For Publish, click the Settings button to display the Publish Settings dialog box with the Flash tab selected. For more information on the Publish Settings dialog box, see "Publishing Flash documents" on page 311.

Using document tabs for multiple documents (Windows only)

When you open multiple documents in Windows, tabs at the top of the Document window identify the open documents and let you easily navigate among them. Tabs appear only when documents are maximized in the Document window.

To make a document active, you click its tab. By default, tabs appear in the order in which the documents were created. You cannot drag tabs to change their order.

To view a document when multiple documents are open:

- Click the document tab.

Saving Flash documents

You can save a Flash FLA document using its current name and location or save the document using a different name or location. You can revert to the last saved version of a document. You can also save Flash MX 2004 content as a Flash MX document.

When a document contains unsaved changes, an asterisk (*) appears after the document name in the document title bar, the application title bar, and the document tab (Windows only). When you save the document, the asterisk is removed.

You can save a document as a template, which lets you use the document as the starting point for a new Flash document (this is similar to how you would use templates in word-processing or web page–editing applications). For information on using templates to create new documents, see "Creating or opening a document and setting properties" on page 12.

When you save a document using the Save command, Flash performs a quick save, which appends new information to the existing file. When you save using the Save As command, Flash arranges the new information into the file, creating a smaller file on disk.

If you quit Flash while one or more documents with unsaved changes are open, Flash prompts you to save the document or documents with the changes.

When you delete items from a document by undoing commands, you can permanently remove the items from the document and reduce the document file size, using the File > Save and Compact command. See "Saving documents when you undo steps" on page 37.

To save a Flash document:

1. Do one of the following:
   - To overwrite the current version on the disk, select File > Save.
   - To save the document in a different location and/or with a different name, or to compress the document, select File > Save As.
2. If you selected the Save As command, or if the document has never been saved before, enter the filename and location.
3. Click Save.

To revert to the last saved version of a document:

- Select File > Revert.

To save a document as a template:

1. Select File > Save As Template.
2. In the Save As Template dialog box, enter a name for the template in the Name text box.
3. Select a category from the Category pop-up menu, or enter a name to create a new category.
4. Enter a description of the template in the Description text box (as many as 255 characters). The description appears when the template is selected in the New Document dialog box.
5. Click OK.

To save a document as a Flash MX document:

1. Select File > Save As.
2. Enter the filename and location.
3. Select Flash MX Document from the Format pop-up menu. If an alert message indicates that content will be deleted if you save in Flash MX format, click Save As Flash MX to continue. This might happen if your document contains features, such as behaviors, that are available only in Flash MX 2004. These features will not be preserved when you save the document in Flash MX format.
4. Click Save.

To save documents when quitting Flash:

1. Select File > Exit (Windows) or Flash > Quit Flash (Macintosh).
2. If you have documents open with unsaved changes, Flash prompts you to save or discard the changes for each document.
   - Click Yes to save the changes and close the document.
   - Click No to close the document without saving the changes.

About adding media content

You can add media content to a Flash document in the Flash authoring environment. You can create vector artwork or text directly in Flash; import vector artwork, bitmaps, video, and sound; and create *symbols*, reusable media content such as buttons.

You can also use ActionScript to add media content to a document dynamically. For more information on ActionScript, see Chapter 2, "ActionScript Basics," in *Using ActionScript in Flash*.

Media content that you add in the authoring environment includes the following:

Vector artwork You can create vector artwork with the Flash drawing and painting tools or import artwork from another application. See Chapter 5, "Drawing," on page 85 and Chapter 7, "Using Imported Artwork," on page 131.

Text You can create *static* text, text whose contents and appearance you determine when you author the document. You can also create *dynamic* text fields, which display text that updates dynamically during runtime, and *input* text fields, which let users enter text for forms or other purposes. See Chapter 6, "Working with Text," on page 105.

Bitmaps You can import bitmaps from other applications, use a bitmap as a file, convert the bitmap to vector artwork, and modify it in other ways. See Chapter 7, "Using Imported Artwork," on page 131.

Video You can import video clips from other applications as embedded or linked files, and select compression and editing options. See Chapter 10, "Working with Video," on page 177.

Sound You can import sound files from other applications and use them as event sounds or streaming sounds in a document. See Chapter 11, "Working with Sound," on page 201.

Symbols You can use symbols, objects that you create once and reuse multiple times. Symbols can be movie clips, buttons, or graphics. Each symbol has its own Timeline. See Chapter 3, "Using Symbols, Instances, and Library Assets," on page 53.

About creating motion and interactivity

Flash provides several ways for you to easily add motion and interactivity to your documents, which creates a compelling user experience. For example, you can make visual elements, such as text, graphics, buttons, or movie clips, move or disappear; you can link to another URL; and you can load another document or movie clip into the current document. The following features let you add motion and interactivity:

Timeline effects are prebuilt animations that you can apply to text, graphics, bitmaps, and buttons, to add motion to visual elements with little effort. See "Using Timeline effects" on page 158.

Tweened and frame-by-frame animation is motion that you create by placing graphics on frames in the Timeline. In tweened animation, you create the beginning and ending frames of the animation, and Flash creates the intermediary frames. In frame-by-frame animation, you create graphics for each frame in the animation. See "Tweened animation" on page 161 and "Frame-by-frame animation" on page 162.

Behaviors are prewritten ActionScript scripts that you add to an object to control that object. Behaviors let you add the power, control, and flexibility of ActionScript coding to your document without having to create the ActionScript code. You can use behaviors to control movie clips and video and sound files. See the following sections:

- "Controlling instances with behaviors" on page 65.
- "Controlling video playback using behaviors" on page 191.
- "Controlling sound playback using behaviors" on page 206.

In screen-based documents, you can use behaviors to control screens. See "Creating controls and transitions for screens with behaviors (Flash Professional only)" on page 228.

Note: You can use ActionScript to create complex or customized interactivity. See Chapter 2, "ActionScript Basics," in *Using ActionScript in Flash*.

About components

Components are movie clips with parameters that let you modify their appearance and behavior. A component can provide a wide range of functionality. A component can be a simple user interface control, such as a radio button or a check box, or it can be a complicated control element, such as a media controller or a scroll pane. A component can even be nonvisual, such as the focus manager that lets you control which object receives focus in an application.

Components let you separate coding and design. They also let you reuse code, and download components created by other developers. For more information, see "Getting Started with Components" in *Using Components*.

Using the library to manage media assets

The library in a Flash document stores media assets that you create or import for use in a Flash document. The library stores imported files such as video clips, sound clips, bitmaps, and imported vector artwork as well as *symbols*. A symbol is a graphic, button, or movie clip that you create once and can reuse multiple times. You can also create a font symbol. For information on symbols, see Chapter 3, "Using Symbols, Instances, and Library Assets," on page 53 and "Creating font symbols" on page 116.

The library also contains components that you have added to your document. Components appear in the library as compiled clips. For more information, see "Components in the Library panel" in *Using Components*.

The Library panel displays a scroll list with the names of all items in the library, which lets you view and organize these elements as you work. An icon next to an item's name in the Library panel indicates the item's file type. The Library panel has an options menu with commands for managing library items.

You can open the library of any Flash document while you are working in Flash, to make the library items from that file available for the current document.

You can create permanent libraries in your Flash application that is available whenever you start Flash. Flash also includes several sample libraries containing buttons, graphics, movie clips, and sounds that you can add to your Flash documents. The sample Flash libraries and permanent libraries that you create are listed in the Window > Common Libraries submenu. For more information, see "Working with common libraries" on page 21.

You can export library assets as a SWF file to a URL to create a runtime-shared library. This lets you link to the library assets from Flash documents that import symbols using runtime sharing. For more information, see "Using shared library assets" on page 69.

To display the Library panel:

- Select Window > Library.

To open the library from another Flash file:

1. Select File > Import > Open External Library.
2. Navigate to the Flash file whose library you want to open and click Open.

   The selected file's library opens in the current document, with the filename at the top of the Library panel. To use items from the selected file's library in the current document, drag the items to the current document's Library panel or to the Stage.

To resize the Library panel:

- Drag the lower right corner of the panel.
- Click the Wide State button to enlarge the Library panel so it shows all the columns.
- Click the Narrow State button to reduce the width of the Library panel.

To change the width of columns:

- Position the pointer between column headers and drag to resize.

  You cannot change the order of columns.

To use the Library options menu:

1. Click the options menu button in the Library panel's title bar to view the options menu.
2. Click an item in the menu.

Working with library items

When you select an item in the Library panel, a thumbnail preview of the item appears at the top of the Library panel. If the selected item is animated or is a sound file, you can use the Play button in the library preview window or the Controller to preview the item. You can use folders in the library to organize library items. See "Working with folders in the Library panel" on page 19.

To use a library item in the current document:

- Drag the item from the Library panel onto the Stage.

  The item is added to the current layer.

To convert an object to a symbol in the library:

- Drag the item from the Stage onto the current Library panel.

To use a library item from the current document in another document:

- Drag the item from the library or Stage into the library or Stage for another document.

Working with folders in the Library panel

You can organize items in the Library panel using folders, much like in the Windows Explorer or the Macintosh Finder. When you create a new symbol, it is stored in the selected folder. If no folder is selected, the symbol is stored at the root of the library.

To create a new folder:

- Click the New Folder button at the bottom of the Library panel.

To open or close a folder:

- Double-click the folder.
- Select the folder and select Expand Folder or Collapse Folder from the Library options menu.

To open or close all folders:

- Select Expand All Folders or Collapse All Folders from the Library options menu.

To move an item between folders:

- Drag the item from one folder to another. If an item with the same name exists in the new location, Flash prompts you to replace the item you are moving.

Sorting items in the Library panel

Columns in the Library panel list the name of an item, its type, the number of times it's used in the file, its linkage status and identifier (if the item is associated with a shared library or is exported for ActionScript), and the date on which it was last modified.

You can sort items in the Library panel alphanumerically by any column. Sorting items lets you view related items together. Items are sorted within folders.

To sort items in the Library panel:

- Click the column header to sort by that column. Click the triangle button to the right of the column headers to reverse the sort order.

Editing items in the library

To edit library items, including imported files, you select options from the Library options menu.

You can also update imported files after editing them in an external editor, using the Update option in the Library options menu. For more information, see "Updating imported files in the Library panel" on page 21.

To edit a library item:

1. Select the item in the Library panel.
2. Select one of the following from the Library options menu:
   - Select Edit to edit an item in Flash.
   - Select Edit With and then select an external application to edit the item.

   Note: When starting a supported external editor, Flash opens the original imported document.

Renaming library items

You can rename items in the library. Changing the library item name of an imported file does not change the filename.

To rename a library item:

- Double-click the item's name and enter the new name in the text box.
- Select the item and select Rename from the Library options menu, and then enter the new name in the text box.
- Right-click (Windows) or Control-click (Macintosh) the item and select Rename from the context menu, and then enter the new name in the text box.

Deleting library items

When you delete an item from the library, all instances or occurrences of that item in the document are also deleted. The Use Count column in the Library panel indicates whether an item is in use.

To delete a library item:

1. Select the item and click the trash can icon at the bottom of the Library panel.
2. In the warning box that appears, select Delete Symbol Instances (the default) to delete the library item and all its instances. Deselect the option to delete only the symbol, which leaves the instances on the Stage.
3. Click Delete.

Finding unused library items

To make organizing a document easier, you can locate unused library items and delete them.

Note: It is not necessary to delete unused library items to reduce a Flash document's file size because unused library items are not included in the SWF file. However, items linked for export are included in the SWF file. For more information, see "Using shared library assets" on page 69.

To find unused library items:

- Select Unused Items from the Library options menu.
- Sort library items by the Use Count column. See "Sorting items in the Library panel" on page 19.

Updating imported files in the Library panel

If you use an external editor to modify files that you have imported into Flash, such as bitmaps or sound files, you can update the files in Flash without reimporting them. You can also update symbols that you have imported from external Flash documents. Updating an imported file replaces its contents with the contents of the external file.

To update an imported file:

- Select the imported file in the Library panel and select Update from the Library options menu.

Working with common libraries

You can use the sample common libraries included with Flash to add buttons or sounds to your documents. You can also create custom common libraries, which you can then use with any documents that you create.

To use an item from a common library in a document:

1. Select Window > Other Panels > Common Libraries, and select a library from the submenu.
2. Drag an item from the common library into the library for the current document.

To create a common library for your Flash application:

1. Create a Flash file with a library containing the symbols that you want to include in the permanent library.
2. Place the Flash file in the Libraries folder located in the Flash application folder on your hard disk.

Note: The Libraries folder is located in the application-level configuration folder, one of several configuration folders placed on your hard drive when you install Flash. For the location of configuration folders, see "Configuration folders installed with Flash" on page 17.

About ActionScript

ActionScript is the Flash scripting language that lets you add complex interactivity, playback control, and data display to a Flash document. You can add ActionScript within the Flash authoring environment using the Actions panel or create external ActionScript files using an external editor.

You don't need to understand every ActionScript element to begin scripting; if you have a clear goal, you can start building scripts with simple actions. You can incorporate new elements of the language as you learn them to accomplish more complicated tasks.

As with other scripting languages, ActionScript follows its own rules of syntax, reserves keywords, provides operators, and lets you use variables to store and retrieve information. ActionScript includes built-in objects and functions and lets you create custom objects and functions. For more information on ActionScript, see Chapter 2, "ActionScript Basics," in *Using ActionScript in Flash*.

ActionScript is based on the ECMAscript specification (ECMA-262), the international standard for the ECMAscript programming language. ActionScript offers a subset of ECMAscript's functionality. For more information about ECMAscript, see the ECMA International website at www.ecma-international.org.

The popular JavaScript language is rooted in the same standard. For this reason, developers who are familiar with JavaScript should find ActionScript immediately familiar and have no trouble learning it quickly.

Multiple Timelines and levels

Flash Player has a stacking order of levels. Every Flash document has a main Timeline located at level 0 in Flash Player. You can use the `loadMovie` action to load other Flash documents (SWF files) into Flash Player at different levels. For more information, see `loadMovie()` in *Flash ActionScript Language Reference*.

If you load documents into levels above level 0, the documents stack on top of one another like drawings on transparent paper; when there is no content on the Stage, you can see through to the content on lower levels. If you load a document into level 0, it replaces the main Timeline. Each document loaded into a level of Flash Player has its own Timeline.

When you add a movie clip instance to a document, the movie clip Timeline is nested inside the main Timeline of the document. You can also nest a movie clip inside another movie clip. For more information, see "Nested movie clips" on page 22.

You can use ActionScript to send a message from one Timeline to another. You must use a target path to specify the location of the Timeline to which you are sending the message. For more information, see "Using absolute and relative target paths" on page 23.

Nested movie clips

Flash documents can have movie clip instances in their Timelines. Each movie clip instance has its own Timeline. You can place a movie clip instance inside another movie clip instance.

Note: A movie clip is a type of symbol. For information on adding movie clips to a document, see Chapter 3, "Using Symbols, Instances, and Library Assets," on page 53.

A movie clip nested inside another movie clip (or inside a document) is a child of that movie clip or document. Relationships between nested movie clips are hierarchical: modifications made to the parent will affect the child. You can use ActionScript to send messages between movie clips and their Timelines. To control a movie clip Timeline from another Timeline, you must specify the location of the movie clip with a target path. In the Movie Explorer, you can view the hierarchy of nested movie clips in a document.

You can also use behaviors, which are ActionScript scripts, to control movie clips. For more information, see "Controlling instances with behaviors" on page 65.

Parent and child movie clips

When you place a movie clip instance on another movie clip's Timeline, the placed movie clip is the child and the other movie clip is the parent. The parent instance contains the child instance. The root Timeline for each level is the parent of all the movie clips on its level, and because it is the topmost Timeline, it has no parent.

A child Timeline nested inside another Timeline is affected by changes made to the parent Timeline. For example, if `portland` is a child of `oregon` and you change the `_xscale` property of `oregon`, then the scale of `portland` also changes.

Timelines can send messages to each other with ActionScript. For example, an action on the last frame of one movie clip can tell another movie clip to play. To use ActionScript to control a Timeline, you must use a target path to specify the location of the Timeline. For more information, see "Writing target paths" on page 25.

Movie clip hierarchy

The parent-child relationships of movie clips are hierarchical. To understand this hierarchy, consider the hierarchy on a computer: the hard disk has a root directory (or folder) and subdirectories. The root directory is analogous to the main Timeline of a Flash document: it is the parent of everything else. The subdirectories are analogous to movie clips.

You can use the movie clip hierarchy in Flash to organize related objects. Any change you make to a parent movie clip also affects its children.

For example, you could create a Flash document containing a car that moves across the Stage. You can use a movie clip symbol to represent the car and set up a motion tween to move it across the Stage.

To add wheels that rotate, you can create a movie clip for a car wheel, and create two instances of this movie clip, named `frontWheel` and `backWheel`. Then you can place the wheels on the car movie clip's Timeline—not on the main Timeline. As children of `car`, `frontWheel` and `backWheel` are affected by any changes made to `car`; they move with the car as it tweens across the Stage.

To make both wheel instances spin, you can set up a motion tween that rotates the wheel symbol. Even after you change `frontWheel` and `backWheel`, they continue to be affected by the tween on their parent movie clip, `car`; the wheels spin, but they also move with the parent movie clip `car` across the Stage.

Using absolute and relative target paths

You can use ActionScript to send messages from one Timeline to another. The Timeline that contains the action is called the *controlling Timeline*, and the Timeline that receives the action is called the *target Timeline*. For example, there could be an action on the last frame of one Timeline that tells another Timeline to play. To refer to a target Timeline, you must use a target path, which indicates the location of a movie clip in the display list.

The following example shows the hierarchy of a document named westCoast on level 0, which contains three movie clips: california, oregon, and washington. Each of these movie clips in turn contains two movie clips.

```
_level0
     westCoast
          california
               sanfrancisco
               bakersfield
          oregon
               portland
               ashland
          washington
               olympia
               ellensburg
```

As on a web server, each Timeline in Flash can be addressed in two ways: with an absolute path or with a relative path. The absolute path of an instance is always a full path from a level name, regardless of which Timeline calls the action; for example, the absolute path to the instance `california` is `_level0.westCoast.california`. A relative path is different when called from different locations; for example, the relative path to `california` from `sanfrancisco` is `_parent`, but from `portland`, it's `_parent._parent.california`.

Absolute paths

An absolute path starts with the name of the level into which the document is loaded and continues through the display list until it reaches the target instance. You can also use the alias `_root` to refer to the topmost Timeline of the current level. For example, an action in the movie clip `california` that refers to the movie clip `oregon` could use the absolute path `_root.westCoast.oregon`.

The first document to open in Flash Player is loaded at level 0. You must assign each additional loaded document a level number. When you use an absolute reference in ActionScript to reference a loaded document, use the form `_levelX`, where *X* is the level number into which the document is loaded. For example, the first document that opens in Flash Player is called `_level0`; a document loaded into level 3 is called `_level3`.

To communicate between documents on different levels, you must use the level name in the target path. The following example shows how the `portland` instance would address the `atlanta` instance located on a movie clip called `georgia` (`georgia` is at the same level as `oregon`):

```
_level5.georgia.atlanta
```

You can use the alias `_root` to refer to the main Timeline of the current level. For the main Timeline, the `_root` alias stands for `_level0` when targeted by a movie clip also on `_level0`. For a document loaded into `_level5`, `_root` is equal to `_level5` when targeted by a movie clip also on level 5. For example, if the movie clips `southcarolina` and `florida` are both loaded into the same level, an action called from the instance `southcarolina` could use the following absolute path to target the instance `florida`:

```
_root.eastCoast.florida
```

Relative paths

A relative path depends on the relationship between the controlling Timeline and the target Timeline. Relative paths can address targets only within their own level of Flash Player. For example, you can't use a relative path in an action on `_level0` that targets a Timeline on `_level5`.

In a relative path, use the keyword `this` to refer to the current Timeline in the current level; use the alias `_parent` to indicate the parent Timeline of the current Timeline. You can use the `_parent` alias repeatedly to go up one level in the movie clip hierarchy within the same level of Flash Player. For example, `_parent._parent` controls a movie clip up two levels in the hierarchy. The topmost Timeline at any level in Flash Player is the only Timeline with a `_parent` value that is undefined.

An action in the Timeline of the instance `charleston`, located one level below `southcarolina`, could use the following target path to target the instance `southcarolina`:

```
_parent
```

To target the instance `eastCoast` (one level up) from an action in `charleston`, you could use the following relative path:

```
_parent._parent
```

To target the instance `atlanta` from an action in the Timeline of `charleston`, you could use the following relative path:

```
_parent._parent.georgia.atlanta
```

Relative paths are useful for reusing scripts. For example, you could attach the following script to a movie clip that magnifies its parent by 150%:

```
onClipEvent (load) {
  _parent._xscale = 150;
  _parent._yscale = 150;
}
```

You can reuse this script by attaching it to any movie clip instance.

Whether you use an absolute or a relative path, you identify a variable in a Timeline or a property of an object with a dot (`.`) followed by the name of the variable or property. For example, the following statement sets the variable `name` in the instance `form` to the value `"Gilbert"`:

```
_root.form.name = "Gilbert";
```

Writing target paths

To control a movie clip, loaded movie, or button, you must specify a target path. In order to specify a target path for a movie clip or button, you must assign an instance name to the movie clip or button. A loaded document doesn't require an instance name because you use its level number as an instance name (for example, `_level5`).

To specify a target path:

- Use the Insert Target Path button (and dialog box) in the Actions panel.
- Enter the target path manually.
- Create an expression that evaluates to a target path. You can use the built-in functions `targetPath` and `eval`.

To assign an instance name:

1. Select a movie clip or button on the Stage.
2. Enter an instance name in the Property inspector.

To insert a target path using the Insert Target Path dialog box:

1. Select the movie clip, frame, or button instance to which you want to assign the action.

   This becomes the controlling Timeline.
2. Select Window > Development Panels > Actions to display the Actions panel if it's not already open.
3. In the Actions toolbox (at the left of the panel), select an action or method that requires a target path.
4. Click the parameter box or location in the script where you want to insert the target path.
5. Click the Insert Target Path button above the Script pane.
6. In the Insert Target Path dialog box, select a syntax: Dots (the default) or Slashes.
7. Select Absolute or Relative for the target path mode.

   For more information, see "Using absolute and relative target paths" on page 23.
8. Select a movie clip in the Insert Target Path display list.
9. Click OK.

To insert a target path manually:

- Follow steps 1–4 and enter an absolute or relative target path in the Actions panel.

To use an expression as a target path:

1. Follow steps 1–3.
2. Do one of the following:
   - Enter an expression that evaluates to a target path in a parameter box.
   - Click to place the insertion point in the script. Then, in the Functions category of the Actions toolbox, double-click the `targetPath` function.

     The `targetPath` function converts a reference to a movie clip into a string.
   - Click to place the insertion point in the script. Then, in the Functions category of the Actions toolbox, select the `eval` function.

   The `eval` function converts a string to a movie clip reference that can be used to call methods such as `play`.

The following script assigns the value 1 to the variable `i`. It then uses the `eval` function to create a reference to a movie clip instance and assigns it to the variable `x`. The variable `x` is now a reference to a movie clip instance and can call the MovieClip object methods.

```
i = 1;
x = eval("mc"+i);
x.play();
// this is equivalent to mc1.play();
```

You can also use the `eval` function to call methods directly, as shown in the following example:

```
eval("mc" + i).play();
```

Working with scenes

To organize a document thematically, you can use scenes. For example, you might use separate scenes for an introduction, a loading message, and credits.

Note: You cannot use scenes in a screen-based document. For information on screens, see Chapter 12, "Working with Screens (Flash Professional Only)," on page 215.

When you publish a Flash document that contains more than one scene, the scenes in the document play back in the order they are listed in the Scene panel in the Flash document. Frames in the document are numbered consecutively through scenes. For example, if a document contains two scenes with ten frames each, the frames in Scene 2 are numbered 11–20.

You can add, delete, duplicate, rename, and change the order of scenes.

To stop or pause a document after each scene, or to let users navigate the document in a nonlinear fashion, you use actions. For more information, see Chapter 2, "ActionScript Basics," in *Using ActionScript in Flash*.

To display the Scene panel:

- Select Window > Design Panels > Scene.

To view a particular scene:

- Select View > Go To and then select the name of the scene from the submenu.

To add a scene:

- Click the Add Scene button in the Scene panel.
- Select Insert > Scene.

To delete a scene:

- Click the Delete Scene button in the Scene panel.

To change the name of a scene:

- Double-click the scene name in the Scene panel and enter the new name.

To duplicate a scene:

- Click the Duplicate Scene button in the Scene panel.

To change the order of a scene in the document:

- Drag the scene name to a different location in the Scene panel.

Using the Movie Explorer

The Movie Explorer provides an easy way for you to view and organize the contents of a document and select elements in the document for modification. It contains a display list of currently used elements, arranged in a navigable hierarchical tree. You can filter which categories of items in the document appear in the Movie Explorer, selecting from text, graphics, buttons, movie clips, actions, and imported files. You can display the selected categories as scenes, symbol definitions, or both. And you can expand and collapse the navigation tree.

The Movie Explorer offers many features to streamline the workflow for creating documents. For example, you can use the Movie Explorer to do the following actions:

- Search for an element in a document by name.
- Familiarize yourself with the structure of a Flash document created by another developer.
- Find all the instances of a particular symbol or action.
- Print the navigable display list that appears in the Movie Explorer.

The Movie Explorer has an options menu as well as a context menu with options for performing operations on selected items or modifying the Movie Explorer display. The options menu is indicated by a check mark with a triangle below it in the title bar of the Movie Explorer.

Note: The Movie Explorer has slightly different functionality when you are working with screens. For more information, see Chapter 12, "Working with Screens (Flash Professional Only)," on page 215.

To view the Movie Explorer:

- Select Window > Other Panels > Movie Explorer.

To filter the categories of items appearing in the Movie Explorer:

- To show text, symbols, ActionScript, imported files, or frames and layers, click one or more of the filtering buttons to the right of the Show option. To customize which items to show, click the Customize button. Select options in the Show area of the Movie Explorer Settings dialog box to view those elements.
- From the options menu in Movie Explorer, select Show Movie Elements to show items in scenes.
- From the options menu in Movie Explorer, select Show Symbol Definitions to show information about symbols.

  Note: Both the Movie Elements option and the Symbol Definitions option can be active at the same time.

To search for an item using the Find text box:

- In the Find text box, enter the item name, font name, ActionScript string, or frame number. The Find feature searches all items that appear in the Movie Explorer.

To select an item in the Movie Explorer:

- Click the item in the navigation tree. Shift-click to select more than one item.

  The full path for the selected item appears at the bottom of the Movie Explorer. Selecting a scene in the Movie Explorer shows the first frame of that scene on the Stage. Selecting an element in the Movie Explorer selects that element on the Stage if the layer containing the element is not locked.

To use the Movie Explorer options menu or context menu commands:

1. Do one of the following:
   - To view the options menu, click the options menu control in the Movie Explorer's title bar.
   - To view the context menu, right-click (Windows) or Control-click (Macintosh) an item in the Movie Explorer navigation tree.
2. Select an option from the menu:

   Go to Location jumps to the selected layer, scene, or frame in the document.

   Go to Symbol Definition jumps to the symbol definition for a symbol that is selected in the Movie Elements area of the Movie Explorer. The symbol definition lists all the files associated with the symbol. (The Show Symbol Definitions option must be selected. See its definition in this list.)

   Select Symbol Instances jumps to the scene containing instances of a symbol that is selected in the Symbol Definitions area of the Movie Explorer. (The Show Movie Elements option must be selected.)

   Find in Library highlights the selected symbol in the document's library (Flash opens the Library panel if it is not already visible).

   Rename lets you enter a new name for a selected element.

   Edit in Place lets you edit a selected symbol on the Stage.

   Edit in New Window lets you edit a selected symbol in a new window.

   Show Movie Elements shows the elements in your document organized into scenes.

   Show Symbol Definitions shows all the elements associated with a symbol.

   Copy All Text to Clipboard copies selected text to the Clipboard. You can paste the text into an external text editor for spell checking or other editing.

   Cut, **Copy**, **Paste**, and **Clear** perform these common functions on a selected element. Modifying an item in the display list modifies the corresponding item in the document.

   Expand Branch expands the navigation tree at the selected element.

   Collapse Branch collapses the navigation tree at the selected element.

   Collapse Others collapses the branches in the navigation tree not containing the selected element.

   Print prints the hierarchical display list that appears in the Movie Explorer.

Using Find and Replace

You can use the Find and Replace feature to find and replace a specified element in a Flash document. You can search for a text string, a font, a color, a symbol, a sound file, a video file, or an imported bitmap file.

You can replace the specified element with another element of the same type. Depending on the type of specified element, there are different options available in the Find and Replace dialog box.

You can find and replace elements in the current document or the current scene. You can search for the next occurrence or all occurrences of an element, and you can replace the current occurrence or all occurrences.

Note: In a screen-based document, you can find and replace elements in the current document or the current screen, but you can't use scenes. For information on working with screens, see Chapter 12, "Working with Screens (Flash Professional Only)," on page 215.

The Live Edit option lets you edit the specified element directly on the Stage. If you use Live Edit when searching for a symbol, Flash opens the symbol in edit-in-place mode.

The Find and Replace Log at the bottom of the Find and Replace dialog box shows the location, name, and type of the elements for which you are searching.

To open the Find and Replace dialog box:

1. Select Edit > Find and Replace.
2. Do one of the following:
   - Select Current Document from the Search In pop-up menu.
   - Select Current Scene from the Search In pop-up menu.

Finding and replacing text

When you find and replace text, you can enter the text string to find and the text string with which to replace it. You can select options for searching by whole word, for matching case, and for selecting which type of text element (text field contents, ActionScript strings, and so on) to include in the search.

To find and replace text:

1. Select Edit > Find and Replace.
2. Select Text from the For pop-up menu.
3. In the Text text box, enter the text that you want to find.
4. In the Replace with Text text box, enter the text that you want to use to replace the existing text.
5. Select options for searching text:

   Whole Word searches for the specified text string as a whole word only, bounded on both sides by spaces, quotes, or similar markers. When Whole Word is deselected, the specified text can be searched as part of a larger word. For example, when Whole Word is deselected, the word *place* can be searched as part of the word *replace*.

Match Case searches for text that exactly matches the case (uppercase and lowercase character formatting) of the specified text when finding and replacing.

Regular Expressions searches for text in regular expressions in ActionScript. An expression is any statement that Flash can evaluate that returns a value. For more information, see ActionScript Reference Guide Help.

Text Field Contents searches the contents of a text field.

Frames/Layers/Parameters searches frame labels, layer names, scene names, and component parameters.

Strings in ActionScript searches strings in ActionScript in the document or scene (external ActionScript files are not searched).

6. Select Live Edit to select the next occurrence of the specified text on the Stage and edit it in place.

   Note: Only the next occurrence is selected for live editing, even if you select Find All in step 6.

7. To find text, do one of the following:
   - Click Find Next to find the next occurrence of the specified text.
   - Click Find All to find all occurrences of the specified text.
8. To replace text, do one of the following:
   - Click Replace to replace the currently selected occurrence of the specified text.
   - Click Replace All to replace all occurrences of the specified text.

Finding and replacing fonts

When you find and replace fonts, you can search or replace by font name, font style, font size, or any combination of those characteristics.

To find and replace fonts:

1. Select Edit > Find and Replace.
2. Select Font from the For pop-up menu, then select from the following options:
   - To search by font name, select Font Name and select a font from the pop-up menu or enter a font name in the text box. When Font Name is deselected, all fonts in the scene or document are searched.
   - To search by font style, select Font Style and select a font style from the pop-up menu. When Font Style is deselected, all font styles in the scene or document are searched.
   - To search by font size, select Font Size and enter a value for minimum and maximum font size to specify the range of font sizes to be searched. When Font Size is deselected, all font sizes in the scene or document are searched.
   - To replace the specified font with a different font name, select Font Name under Replace With and select a font name from the pop-up menu or enter a name in the text box. When Font Name is deselected under Replace with, the current font name remains unchanged.

- To replace the specified font with a different font style, select Font Style under Replace With and select a font style from the pop-up menu. When Font Style is deselected under Replace with, the current style of the specified font remains unchanged.
- To replace the specified font with a different font size, select Font Size under Replace With and enter values for minimum and maximum font size. When Font Size is deselected under Replace With, the current size of the specified font remains unchanged.

3. Select Live Edit to select the next occurrence of the specified font on the Stage and edit it in place.

   Note: Only the next occurrence is selected for live editing, even if you select Find All in step 4.

4. To find a font, do one of the following:
   - Click Find Next to find the next occurrence of the specified font.
   - Click Find All to find all occurrences of the specified font.
5. To replace a font, do one of the following:
   - Click Replace to replace the currently selected occurrence of the specified font.
   - Click Replace All to replace all occurrences of the specified font.

Finding and replacing colors

To find and replace a color, you can select a color to find or replace by picking a color swatch in the color pop-up window, by entering a hexadecimal color value in the color pop-up window, by using the system color picker, or by selecting a color from the desktop with the eyedropper tool. You can find and replace a color in a stroke, in a fill, in text, or in any combination of those items.

You cannot find and replace colors in grouped objects.

Note: To find and replace colors in a GIF or JPEG file in a Flash document, edit the file in Macromedia Fireworks or a similar image-editing application.

To find and replace a color:

1. Select Edit > Find and Replace.
2. Select Color from the For pop-up menu.
3. To search for a color, click the Color control and do one of the following:
   - Select a color swatch from the color pop-up window.
   - Enter a hexadecimal color value in the Hex Edit text box in the color pop-up window.
   - Click the Color Picker button and select a color from the system color picker.
   - Drag from the Color control to make the eyedropper tool appear. Select any color on your screen.
4. To select a color to use in replacing the specified color, click the Color control under Replace With and do one of the following:
   - Select a color swatch from the color pop-up window.
   - Enter a hexadecimal color value in the Hex Edit text box in the color pop-up window.

- Click the Color Picker button and select a color from the system color picker.
- Drag from the Color control to make the eyedropper tool appear. Select any color on your screen.

5. Select the Fills, Strokes, or Text option or any combination of those options to specify which occurrence of the color to find and replace.
6. Select Live Edit to select the next occurrence of the specified color on the Stage and edit it in place.

   Note: Only the next occurrence is selected for live editing, even if you select Find All in step 6.

7. To find a color, do one of the following:
   - Click Find Next to find the next occurrence of the specified color.
   - Click Find All to find all occurrences of the specified color.
8. To replace a color, do one of the following:
   - Click Replace to replace the currently selected occurrence of the specified color.
   - Click Replace All to replace all occurrences of the specified color.

Finding and replacing symbols

When you find and replace symbols, you can search for a symbol by name. You can replace a symbol with another symbol of any type—movie clip, button, or graphic.

To find and replace a symbol:

1. Select Edit > Find and Replace.
2. Select Symbol from the For pop-up menu.
3. For Name, select a name from the pop-up menu.
4. Under Replace With, for Name select a name from the pop-up menu.
5. Select Live Edit to select the next occurrence of the specified symbol on the Stage and edit it in place.

   Note: Only the next occurrence is selected for editing, even if you select Find All in step 5.

6. To find a symbol, do one of the following:
   - Click Find Next to find the next occurrence of the specified symbol.
   - Click Find All to find all occurrences of the specified symbol.
7. To replace a symbol, do one of the following:
   - Click Replace to replace the currently selected occurrence of the specified symbol.
   - Click Replace All to replace all occurrences of the specified symbol.

Finding and replacing sound, video, or bitmap files

When you find and replace a sound, video, or bitmap file, you can search for the file by name. You can replace the file with another file of the same type. That is, you can replace a sound with a sound, a video with a video, or a bitmap with a bitmap.

To find and replace a sound, video, or bitmap:

1. Select Edit > Find and Replace.
2. Select Sound, Video, or Bitmap from the For pop-up menu.
3. For Name, enter a sound, video, or bitmap filename or select a name from the pop-up menu.
4. Under Replace With, for Name enter a sound, video, or bitmap filename or select a name from the pop-up menu.
5. Select Live Edit to select the next occurrence of the specified sound, video, or bitmap on the Stage and edit it in place.

   Note: Only the next occurrence is selected for editing, even if you select Find All in step 5.

6. To find a sound, video, or bitmap, do one of the following:
   - Click Find Next to find the next occurrence of the specified sound, video, or bitmap.
   - Click Find All to find all occurrences of the specified sound, video, or bitmap.
7. To replace a sound, video, or bitmap, do one of the following:
   - Click Replace to replace the currently selected occurrence of the specified sound, video, or bitmap.
   - Click Replace All to replace all occurrences of the specified sound, video, or bitmap.

Using the Undo, Redo, and Repeat menu commands

The Edit > Undo and Edit > Redo commands let you undo and redo steps as you work on Flash documents. The names of the Undo and Redo commands change to reflect the last step performed.

To remove deleted items from a document after using the Undo command, you use the Save and Compact command. See "Saving documents when you undo steps" on page 37.

You can use the Repeat command to reapply a step to the same object or to a different object. For example, if you move a shape named shape_A, you can select Edit > Repeat to move the shape again, or you can select another shape, shape_B, and select Edit > Repeat to move the second shape by the same amount.

By default, Flash supports 100 levels of undo for the Undo menu command. You can select the number of undo and redo levels, from 2 to 9999, in Flash Preferences. For more information, see "Setting preferences in Flash" in *Getting Started with Flash*.

To undo a step:

- Select Edit > Undo.

To redo a step:

- Select Edit > Redo.

To repeat a step:

- With an object selected on the Stage, select Edit > Repeat.

Using the History panel

The History panel shows a list of the steps you've performed in the active document since you created or opened that document, up to a specified maximum number of steps. (The History panel doesn't show steps you've performed in other documents.) The slider in the History panel initially points to the last step that you performed.

You can use the History panel to undo or redo individual steps or multiple steps at once. You can apply steps from the History panel to the same object or to a different object in the document. However, you cannot rearrange the order of steps in the History panel. The History panel is a record of steps in the order in which they were performed.

Note: If you undo a step or a series of steps and then do something new in the document, you can no longer redo the steps in the History panel; they disappear from the panel.

To remove deleted items from a document after you undo a step in the History panel, you use the Save and Compact command. For more information, see "Saving documents when you undo steps" on page 37.

By default, Flash supports 100 levels of undo for the History panel. You can select the number of undo and redo levels, from 2 to 9999, in Flash Preferences. For more information, see "Setting preferences in Flash" in *Getting Started with Flash*.

You can clear the History panel to erase the history list for the current document. After clearing the history list, you cannot undo the steps that are cleared. Clearing the history list does not undo steps; it merely removes the record of those steps from the current document's memory.

Closing a document clears its history. If you know you want to use steps from a document after that document is closed, copy the steps with the Copy Steps command or save the steps as a command. For more information, see "Copying and pasting steps between documents" on page 36 or "Automating tasks with the Commands menu" on page 37.

To open the History panel:

- Select Window > Other Panels > History.

To erase the history list for the current document:

1. In the History panel options menu, select Clear History.
2. Click Yes to confirm the Clear command.

Undoing steps with the History panel

You can undo the last step or multiple steps with the History panel. When you undo a step, the step is dimmed in the History panel.

To undo the last step performed:

- Drag the History panel slider up one step in the list.

To undo multiple steps at once:

- Drag the slider to point to any step.
- Click to the left of a step along the path of the slider; the slider scrolls automatically to that step, undoing all subsequent steps as it scrolls.

  Note: Scrolling to a step (and selecting the subsequent steps) is different from selecting an individual step. To scroll to a step, you must click to the left of the step.

Replaying steps with the History panel

You can replay individual steps or multiple steps using the History panel.

When you replay steps with the History panel, the steps that play are the steps that are selected (highlighted) in the History panel, not necessarily the step currently indicated by the slider.

You can apply steps in the History panel to any selected object in the document.

To replay one step:

- In the History panel, select a step and click the Replay button. The step replays and a copy of it appears in the History panel.

To replay a series of adjacent steps:

1. Select steps in the History panel by doing one of the following:
   - Drag from one step to another. (Don't drag the slider; just drag from the text label of one step to the text label of another step.)
   - Select the first step, then Shift-click the last step; or select the last step and then Shift-click the first step.
2. Click Replay.

   The steps replay in order, and a new step, labeled Replay Steps, appears in the History panel.

To replay nonadjacent steps:

1. Select a step in the History panel, and Control-click (Windows) or Command-click (Macintosh) other steps.

   You can also Control-click or Command-click to deselect a selected step.
2. Click Replay.

   The selected steps replay in order, and a new step, labeled Replay Steps, appears in the History panel.

Copying and pasting steps between documents

Each open document has its own history of steps. You can copy steps from one document and paste them into another, using the Copy Steps command in the History panel options menu. If you copy steps into a text editor, the steps are pasted as JavaScript code.

To reuse steps from one document in another document:

1. In the document containing the steps you want to reuse, select the steps in the History panel.
2. In the History panel options menu, select Copy Steps.
3. Open the document into which you want to paste the steps.
4. Select an object to which you want to apply the steps.
5. Select Edit > Paste to paste the steps.

   The steps play back as they're pasted into the document's History panel. The History panel shows them as only one step, called Paste Steps.

Saving documents when you undo steps

By default, when you undo a step using Edit > Undo or the History panel, the file size of the Flash document does not change, even if you delete an item in the document. For example, if you import a video file into a document, and then undo the import, the file size of the document still includes the size of the video file. This is because any items that you delete from a document when performing an Undo command are preserved in case you want to restore the items with a Redo command. You can permanently remove the deleted items from the document, and reduce the document file size, by using the Save and Compact command.

To permanently remove items deleted by the Undo command:

- Select File > Save and Compact.

Automating tasks with the Commands menu

When you create documents, you might want to perform the same task numerous times. You can create a new command in the Commands menu from steps in the History panel and reuse the command multiple times. Steps replay exactly as they were originally performed. You can't modify the steps as you replay them.

You should create and save a new command if there's a chance you might want to use a set of steps again, especially if you want to use those steps the next time you start Flash. Saved commands are retained permanently, unless you delete them. Steps that you copy using the History panel Copy Steps command are discarded when you copy something else. For more information, see "Copying and pasting steps between documents" on page 36.

About steps that can't be used in commands

Some tasks in Flash can't be saved as commands or repeated using the Edit > Repeat menu item. These commands can be undone and redone, but they cannot be repeated.

Examples of actions that can't be saved as commands or repeated include selecting a frame or modifying a document size. If you attempt to save an unrepeatable action as a command, the command is not saved.

Creating and managing commands

You can create a command from selected steps in the History panel. In the Manage Saved Commands dialog box, you can rename or delete commands.

To create a command:

1. Select a step or set of steps in the History panel.
2. Select Save As Command from the History panel options menu.
3. Enter a name for the command and click OK.

   The command appears in the Commands menu.

   Note: The command is saved as a JavaScript file (with the extension .jsfl) in your Flash MX 2004\<language>\First Run\Commands folder.

To edit the names of commands in the Commands menu:

1. Select Commands > Edit Command List.
2. Select a command to rename and enter a new name for it.
3. Click Close.

To delete a name from the Commands menu:

1. Select Commands > Edit Command List.
2. Select a command.
3. Click Delete, and click Close.

Running commands

You can use the commands that you create by selecting the command name from the Commands menu.

You can also run commands that are available on your system as JavaScript or Flash JavaScript files.

To use a saved command:

- Select the command from the Commands menu.

To run a JavaScript or Flash JavaScript command:

1. Select Commands > Run Command.
2. Navigate to the script that you want to run, and click Open.

Getting more commands

You can use the Get More Commands option in the Commands menu to link to the Flash Exchange website at www.macromedia.com/cfusion/exchange/index.cfm and download commands that other Flash users have posted. For more information on the commands posted there, see Flash Exchange.

To get more commands:

1. Make sure you are connected to the Internet.
2. Select Commands > Get More Commands.

About customizing context menus in Flash documents

You can customize the standard context menu and the text-editing context menu that appears with Flash documents in Flash Player 7.

- The standard context menu appears when a user right-clicks (Windows) or Control-clicks (Macintosh) on a document in Flash Player, in any area except an editable text field. You can add custom items to the menu, and hide any built-in items in the menu except Settings and Debugger.
- The editing context menu appears when a user right-clicks (Windows) or Control-clicks (Macintosh) in an editable text field in a document in Flash Player. You can add custom items to this menu. You cannot hide any built-in items.

Note: Flash Player also displays an error context menu when a user right-clicks (Windows) or Control-clicks (Macintosh) in Flash Player and no document is loaded. You cannot customize this menu.

You customize context menus in Flash Player 7 using the `contextMenu` and `contextMenuItem` objects in ActionScript. For more information on using these objects, see "ContextMenu class" in *Flash ActionScript Language Reference*.

Remember the following criteria when creating custom context menu items for Flash Player:

- Custom items are added to a context menu in the order in which they are created. You cannot modify this order after the items are created.
- You can specify the visibility and enabling of custom items.
- Custom context menu items are automatically encoded using Unicode UTF-8 text encoding.

About the links menu in Flash Player

If a user is using a Netscape browser or an Active X application to display Flash Player, the player displays a links menu for all Flash documents. If the user right-clicks (Windows) or Control-clicks (Macintosh) on a text link in the Flash document, the links menu appears with the following menu items:

Open opens the link.

Open in New Window opens the link in a new window.

Copy Link copies the link to the Clipboard.

In addition, the user can open a link in a new window by doing the following:

- In a Windows Netscape browser: Control-click the link.
- In a Macintosh Netscape browser: Command-click the link.
- In an Active X application: Shift-click the link.

Speeding up document display

To speed up the document display, you can use commands in the View menu to turn off rendering-quality features that require extra computing and slow down document display.

None of these commands have any effect on how Flash exports a document. To specify the display quality of Flash documents in a web browser, you use the `object` and `embed` parameters. The Publish command can do this for you automatically. For more information, see "Publishing Flash documents" on page 311.

To change the document display speed:

- Select View > Preview Mode, and select from the following options:

  Outlines displays only the outlines of the shapes in your scene and causes all lines to appear as thin lines. This makes it easier to reshape your graphic elements and to display complex scenes quickly.

  Fast turns off anti-aliasing and displays all the colors and line styles of your drawing.

  Antialias turns on anti-aliasing for lines, shapes, and bitmaps. It displays shapes and lines so that their edges appear smoother on the screen. This option draws more slowly than the Fast option. Anti-aliasing works best on video cards that provide thousands (16-bit) or millions (24-bit) of colors. In 16- or 256-color mode, black lines are smoothed, but colors might look better in Fast mode.

  Antialias Text smooths the edges of any text. This command works best with large font sizes and can be slow with large amounts of text. This is the most common mode in which to work.

  Full renders all content on the Stage fully. This setting may slow down display.

Optimizing Flash documents

As your document file size increases, so does its download time and playback speed. You can take several steps to prepare your document for optimal playback. As part of the publishing process, Flash automatically performs some optimization on documents: for example, it detects duplicate shapes on export and places them in the file only once, and it converts nested groups into single groups.

Before exporting a document, you can optimize it further by using various strategies to reduce the file size. You can also compress a SWF file as you publish it. (See Chapter 15, "Publishing," on page 309.) As you make changes, it's a good idea to test your document by running it on a variety of computers, operating systems, and Internet connections.

To optimize documents:

- Use symbols, animated or otherwise, for every element that appears more than once.
- When creating animation sequences, use tweened animations, whenever possible. These animations use less file space than a series of keyframes.
- For animation sequences, use movie clips instead of graphic symbols.
- Limit the area of change in each keyframe; make the action take place in as small an area as possible.

- Avoid animating bitmap elements; use bitmap images as background or static elements.
- For sound, use MP3, the smallest sound format, whenever possible.

To optimize elements and lines:

- Group elements as much as possible.
- Use layers to separate elements that change during the animation from those that do not.
- Use Modify > Curves > Optimize to minimize the number of separate lines that are used to describe shapes.
- Limit the number of special line types, such as dashed, dotted, ragged, and so on. Solid lines require less memory. Lines created with the Pencil tool require less memory than brush strokes.

To optimize text and fonts:

- Limit the number of fonts and font styles. Use embedded fonts sparingly because they increase file size.
- For Embed Fonts options, select only the characters needed instead of including the entire font.

To optimize colors:

- Use the Color menu in the symbol Property inspector to create many instances of a single symbol in different colors.
- Use the Color Mixer (Window > Color Mixer) to match the color palette of the document to a browser-specific palette.
- Use gradients sparingly. Filling an area with gradient color requires about 50 bytes more than filling it with solid color.
- Use alpha transparency sparingly because it can slow playback.

Testing document download performance

Flash Player attempts to meet the frame rate you set; the actual frame rate during playback can vary on different computers. If a document that is downloading reaches a particular frame before the frame's required data has downloaded, the document pauses until the data arrives.

To view downloading performance graphically, you can use the Bandwidth Profiler, which shows how much data is sent for each frame according to the modem speed you specify. The Bandwidth Profiler is divided into two panes. The left pane shows information about the document, the download settings, the state, and streams, if any are included. The right pane shows information about individual frames in the document.

In simulating the downloading speed, Flash uses estimates of typical Internet performance, not the exact modem speed. For example, if you select to simulate a modem speed of 28.8 Kbps, Flash sets the actual rate to 2.3 Kbps to reflect typical Internet performance. The profiler also compensates for the added compression support for SWF files, which reduces the file size and improves streaming performance.

When external SWF files, GIF and XML files, and variables are streamed into a player by using ActionScript calls such as `loadMovie` and `getUrl`, the data flows at the rate set for streaming. The stream rate for the main SWF file is reduced based on the reduction of bandwidth caused by the additional data requests. It's helpful to test your document at each speed and on each computer that you plan to support. This helps you ensure that the document doesn't overburden the slowest connection and computer for which it is designed.

You can also generate a report of frames that are slowing playback and then optimize or eliminate some of the content in those frames. For more information, see "Optimizing Flash documents" on page 40.

To change the settings for the SWF file created using the Test Movie and Test Scene commands, use File > Publish Settings. For more information, see "Publishing Flash documents" on page 311.

To test download performance:

1. Do one of the following:
   - Select Control > Test Scene or Control > Test Movie.

     If you test a scene or document, Flash publishes the current selection as a SWF file using the settings in the Publish Settings dialog box. (See "Publishing Flash documents" on page 311.) The SWF file opens in a new window and begins playing immediately.
   - Select File > Open, and select a SWF file.
2. Select View > Download Settings, and select a download speed to determine the streaming rate that Flash simulates: 14.4 Kbps, 28.8 Kbps, 56 Kbps, DSL, T1 or a user setting. To enter a custom user setting, select Customize.
3. When viewing the SWF file, select View > Bandwidth Profiler to show a graph of the downloading performance.

   The left side of the profiler displays information about the document, its settings, its state, and streams, if any are included in the document.

   The right section of the profiler shows the Timeline header and graph. In the graph, each bar represents an individual frame of the document. The size of the bar corresponds to that frame's size in bytes. The red line beneath the Timeline header indicates whether a given frame streams in real time with the current modem speed set in the Control menu. If a bar extends above the red line, the document must wait for that frame to load.
4. Select View > Simulate Download to turn streaming off or on.

   If you turn streaming off, the document starts over without simulating a web connection.
5. Click a bar on the graph to show settings for the corresponding frame in the left window and stop the document.

6. If necessary, adjust the view of the graph by taking one of the following actions:
   - Select View > Streaming Graph to show which frames cause pauses.

     This default view displays alternating light and dark gray blocks that represent each frame. The side of each block indicates its relative byte size. The first frame stores a symbol's contents, so it is often larger than other frames.
   - Select View > Frame by Frame Graph to display the size of each frame.

     This view helps you see which frames contribute to streaming delays. If any frame block extends above the red line in the graph, Flash Player stops playback until the entire frame downloads.
7. Close the test window to return to the authoring environment.

After you set up a test environment using the Bandwidth Profiler, you can open any SWF file directly in test mode. The file opens in a Flash Player window, using the Bandwidth Profiler and other selected viewing options.

For more information on debugging your documents, see Chapter 4, "Writing and Debugging Scripts" in *Using ActionScript in Flash*.

To generate a report listing the amount of data in the final Flash Player file:

1. Select File > Publish Settings, and click the Flash tab.
2. Select Generate Size Report.
3. Click Publish.

   Flash generates a text file with the extension .txt. (If the document file is myMovie.fla, the text file is myMovie Report.txt.) The report lists the size of each frame, shape, text, sound, video and ActionScript script by frame.

Printing from the Flash authoring tool

You can print frames from Flash documents as you work, to preview and edit your documents.

You can also specify frames to be printable from Flash Player by a viewer that shows the Flash document. See Chapter 18, "Printing from SWF Files," on page 375.

When printing frames from a Flash document, you use the Print dialog box to specify the range of scenes or frames you want to print as well as the number of copies. In Windows, the Page Setup dialog box specifies paper size, orientation, and various print options—including margin settings and whether all frames are to be printed for each page. On the Macintosh, these options are divided between the Page Setup and the Print Margins dialog boxes.

The Print and Page Setup dialog boxes are standard within either operating system, and their appearance depends on the selected printer driver.

To set printing options:

1. Select File > Page Setup (Windows) or File > Print Margins (Macintosh).
2. Set page margins. Select both Center options to print the frame in the center of the page.
3. In the Frames pop-up menu, select whether to print all frames in the document or only the first frame of each scene.
4. In the Layout pop-up menu, select from the following options:

   Actual Size prints the frame at full size. Enter a value for Scale to reduce or enlarge the printed frame.

   Fit on One Page reduces or enlarges each frame so it fills the print area of the page.

   Storyboard options print several thumbnails on one page. You can select from Boxes, Grid, or Blank. Enter the number of thumbnails per page in the Frames text box. Set the space between the thumbnails in the Story Margin text box. Select Label to print the frame label as a thumbnail.

To print frames:

- Select File > Print.

CHAPTER 2
Working with Projects (Flash Professional Only)

In Macromedia Flash MX Professional 2004, you can use Flash Projects to manage multiple document files in a single project. Flash Projects allow you to group multiple, related files together to create complex applications.

You can use version-control features with projects to ensure that the correct file versions are used during editing, and to prevent accidental overwriting. To use version control, you must first add files to a project. For information on version control, see "Using version control with projects (Flash Professional only)" on page 49.

Flash Projects include the following features:

- A Flash Project can contain any Flash or other file type, including previous versions of FLA and SWF files.
- You can add an existing file to a Flash Project. Each file can be added to a particular Flash Project only once. Files can be organized in nested folders.
- A Flash Project is an XML file with the file extension .flp–for example, myProject.flp. The XML file references all the document files contained in the Flash Project.
- A Flash Project can contain another Flash Project (FLP file).
- Changes that you make to a project are updated to the FLP file immediately, so the file is always current. (You do not need to do a Save File operation.)
- You can create a Flash Project in the Flash MX Professional 2004 authoring environment, or you can create the XML file for a Flash Project in an external application.
- Flash Projects use UTF-8 text encoding. All filenames and folder names in a Flash Project must be UTF-8 compatible.

This chapter contains the following sections:

Creating and managing projects (Flash Professional only)

You use the Flash Project panel to create and manage projects.The panel displays the contents of a Flash Project in a collapsible tree structure. The panel title bar displays the project name.

If a project file is missing (not in its specified location), a Missing File icon appears next to the filename. You can search for a missing file or delete the file from the project.

When you publish a project, each FLA file in the project is published with the publish profile specified for that file. You should specify the publish profiles in the Project Settings dialog box before you publish a project.

Only one project can be open at one time. If a project is open and you open or create another project, Flash automatically saves and closes the first file.

To view the Flash Project panel:

- Select Window > Project.

To view the Project pop-up menu:

- When a project is open, click the Project button at the upper left corner of the Flash Project panel.

To create a new project:

1. Do one of the following to open a new project:
   - Select New Project from the Project pop-up menu.
   - If no other project is open, open the Flash Project panel and select Create a New Project in the panel window.
   - Select File > New. On the General tab, select Flash Project.
   - If no project is currently open, right-click (Windows) or Control-click (Macintosh) in the Document window of a saved Flash document or ActionScript file and select Add to New Project from the context menu.
2. In the New Project dialog box, enter a name for the project and click Save.

To open an existing project, do one of the following:

- Select Open Project from the Project pop-up menu. Navigate to the project and click Open.
- Double-click the file.
- If no other project is open, open the Flash Project panel and select Open an Existing Project in the panel window. Navigate to the project and click Open.
- Select File > Open. Navigate to the project and click Open.

To add a file, do one of the following:

- Click the Add Files (+) button at the lower right corner of the Flash Project panel. Select one or more files and click Add.
- Right-click (Windows) or Control-click (Macintosh) in the Document window of an open FLA or AS file and select Add to Project from the context menu.

Note: A file must be saved before you can add it to a project. You can add a file to a given project only once. If you attempt to add a file to the same project more than once, Flash displays an error message.

To create a folder:

1. Click the Folder button at the lower right corner of the Flash Project panel.
2. Enter a name for the folder and click OK.

Note: Folders at the same level on the same branch of the project tree structure must have unique names. If there is a folder name conflict, Flash displays an error message.

To move a file or folder:

- Drag the file or folder to a new location in the project tree structure. When you move a folder, all of its contents are moved.

Note: If you drag a folder to a location with another folder of the same name, Flash merges the contents of the two folders in the new location.

To delete a file or folder, select the item in the Flash Project panel and do one of the following:

- Click the Remove button at the lower right corner of the Flash Project panel.
- Press the Delete key.
- Right-click (Windows) or Control-click (Macintosh) the file or folder and select Remove from the context menu.

To open a file from the Flash Project panel in Flash:

- Double-click the filename in the Flash Project panel.

  If the file is of a native file type (a type supported by the Flash authoring tool), the file opens in Flash. If it is nonnative file type, the file opens in the application used to create it.

To test a project:

1. Click Test Project in the Flash Project panel.
2. If the project contains no FLA, HTML, or HTM file, Flash displays an error message. Click OK and add a file of the appropriate type.
3. If no FLA, HTML, or HTM file is designated as the default document, Flash displays an error message. Click OK. In the Select Default Document dialog box, select a document and click OK.

   When a default document is present, the Test Project feature publishes all FLA files in the document. If the default document is a FLA file, the Test Movie command is executed. If it is an HTML file, a browser is opened.

To specify a publish profile for a FLA file in a project:

1. Select the file in the Flash Project panel and do one of the following:
   - Select Settings from the Project pop-up menu.
   - Right-click (Windows) or Control-click (Macintosh) and select Settings from the context menu.
2. In the Project Settings dialog box, select the FLA file in the tree structure.
3. Select a publish profile from the Profile menu. For information on publish profiles, see "Using publish profiles" on page 327.

To publish a project:

- Select Publish Project from the Project pop-up menu.

Note: Flash uses default publish profiles for publishing FLA files in the project, unless you select other profiles. See the procedure above for selecting publish profiles.

To save files in a project when testing or publishing:

1. Select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Editing tab.
2. Under Project Preferences, click Save Project Files on Test Project or Publish Project.

   When this option is selected, Flash saves all open files in the current project before executing the Test Project or Publish Project operation.

To close a project:

- Select Close Project from the Project pop-up menu.

  By default, Flash closes all files in a project when you close the project. To change this behavior, deselect the Close Open Files on Project Close option in Editing Preferences.

To close all files when you close a project:

1. Select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Editing tab.
2. Under Project Preferences, click Close Open Files on Project Close (selected by default).

   When this option is selected, Flash closes all open files in the current project when the project is closed.

To rename a project or a folder:

1. Select the project name or folder name in the Flash Project panel and do one of the following:
   - Select Rename from the Project pop-up menu.
   - Right-click (Windows) or Control-click (Macintosh) the item and select Rename from the context menu.
2. Enter a new name and click OK.

Note: By default, a project is given the same name as the first file added to the project. To rename a project, you must use the Rename menu item. Renaming the FLP file for a project does not rename the project.

To find a missing file:

1. Select the filename in the Flash Project panel.
2. Do one of the following:
   - Select Find Missing File from the Project pop-up menu.
   - Right-click (Windows) or Control-click (Macintosh) and select Find Missing File from the context menu.
3. Navigate to the file and click OK.

Using version control with projects (Flash Professional only)

Version control in Flash MX Professional 2004 lets you ensure that each author working in a project file is always using the latest version of a file, and that multiple authors do not overwrite each other's work.

To use version-control features, you must define a site for the project. You can specify a local, network, or FTP connection, or you can specify custom plug-ins for version control systems. If you experience problems when setting up a remote site, see "Troubleshooting remote folder setup (Flash Professional only)" on page 51.

On Windows, you can use Flash projects with SourceSafe. You must have Microsoft Visual SourceSafe Client version 6 installed.

To define a site for version control:

1. Create a new project and add files. See "Creating and managing projects (Flash Professional only)" on page 46.
2. Select File > Edit Sites.
3. In the Edit Sites dialog box, click New.
4. In the Site Definition dialog box, enter the site name, the local root path, and the e-mail address and name of the user.
5. To specify a local, network, or FTP connection, select Local/Network or FTP from the Connection menu. Enter the location information for the Local/Network path or for the FTP connection and skip the next step.

6. To specify a Visual SourceSafe database, select SourceSafe Database from the Connection menu.

   Note: SourceSafe database support is available for Windows only. You must have Microsoft Visual SourceSafe Client version 6 installed.

   a. In the Database Path text box, click Browse to browse for the VSS database you want, or enter the full file path. The file you select becomes the srcsafe.ini file used to initialize SourceSafe.

   b. In the Project text box, enter the project within the VSS database you want to use as the remote site's root directory.

   c. In the Username and Password text boxes, enter your login user name and password for the selected database. If you don't know your username and password, check with your system administrator.

   d. Click OK to return to the Site Definition dialog box.

7. In the Flash Project panel (Window > Project), select Settings from the Project pop-up menu or context menu.

8. In the Project Settings dialog box, select the site definition from the Site menu in the Version Control section. Click OK.

9. In the Project pop-up menu, select Check In. Flash checks all files in the current project into the site.

To edit a file with version control applied:

1. Open the project that contains the file, as described in "Creating and managing projects (Flash Professional only)" on page 46.

2. Select the file in the tree structure in the project panel and select Check Out from the project context menu.

   The icon next to the filename in the tree structure indicates that the file is checked out.

3. To check a file back in, select the file in the project panel and select Check In from the project context menu.

   The icon next to the filename in the tree structure indicates that the file is checked in.

To open a file from a version-control site:

1. Select File > Open from Site.

2. In the Open from Site dialog box, select the site from the Site menu.

3. Select the file in the site.

4. If the file exists on your local system, Flash displays a message indicating whether the file is checked out and, if so, asking whether you want to overwrite it. Click Yes to overwrite the local version with the version from the remote site.

Troubleshooting remote folder setup (Flash Professional only)

A web server can be configured in a wide variety of ways. The following list provides information on some common issues you may encounter in setting up a remote folder for use with version control, and how to resolve them:

- The Flash FTP implementation may not work properly with certain proxy servers, multilevel firewalls, and other forms of indirect server access. If you encounter problems with FTP access, ask your local system administrator for help.
- For the Flash FTP implementation, you must connect to the remote system's root folder. (In many applications, you can connect to any remote directory, then navigate through the remote file system to find the directory you want.) Be sure that you indicate the remote system's root folder as the host directory.
- If you have problems connecting, and you've specified the host directory using a single slash (/), you might need to specify a relative path between the directory you are connecting to and the remote root folder. For example, if the remote root folder is a higher-level directory, you may need to specify a ../../ for the host directory.
- Filenames and folder names that contain spaces and special characters often cause problems when transferred to a remote site. Use underscores in place of spaces, and avoid special characters in filenames and folder names wherever possible. In particular, colons, slashes, periods, and apostrophes in filenames or folder names can cause problems.
- If problems persist, try uploading with an external FTP program to find out if the problem is specific to using FTP in Flash.

CHAPTER 3
Using Symbols, Instances, and Library Assets

A *symbol* is a graphic, button, or movie clip that you create in Macromedia Flash MX 2004 or Macromedia Flash MX Professional 2004.

You create the symbol only once; you can then reuse it throughout your document or in other documents. A symbol can include artwork that you import from another application. Any symbol that you create automatically becomes part of the library for the current document. For more information on the library, see "Using the library to manage media assets" on page 17.

This chapter describes how to create symbols and instances in the Flash authoring environment. You can also create buttons, movie clips, and graphics using the Button Class and MovieClip Class (use the drawing methods of the MovieClip class to create graphics). See "Button class" and "MovieClip class"in *Flash ActionScript Language Reference*.

When you create a symbol in the authoring environment, each symbol has its own Timeline. You can add frames, keyframes, and layers to a symbol Timeline, just as you can to the main Timeline. For more information, see "Using the Timeline" in *Getting Started with Flash*. If the symbol is a movie clip or a button, you can control the symbol with ActionScript. For more information, see Chapter 5, "Handling Events" in *Using ActionScript in Flash*.

An *instance* is a copy of a symbol located on the Stage or nested inside another symbol. An instance can be very different from its symbol in color, size, and function. Editing the symbol updates all of its instances, but applying effects to an instance of a symbol updates only that instance.

Using symbols in your documents dramatically reduces file size; saving several instances of a symbol requires less storage space than saving multiple copies of the contents of the symbol. For example, you can reduce the file size of your documents by converting static graphics, such as background images, into symbols and then reusing them. Using symbols can also speed SWF file playback, because a symbol needs to be downloaded to Flash Player only once.

You can share symbols among documents as shared library assets during authoring or at runtime. For runtime shared assets, you can link assets in a source document to any number of destination document, without importing the assets into the destination document. For assets shared during authoring, you can update or replace a symbol with any other symbol available on your local network. See "Using shared library assets" on page 69.

If you import library assets that have the same name as assets already in the library, you can resolve naming conflicts without accidentally overwriting existing assets. See "Resolving conflicts between library assets" on page 72.

For an introduction to using symbols and instances, select Help > How Do I > Quick Tasks > Create Symbols and Instances.

This chapter contains the following sections:

Types of symbols

Each symbol has a unique Timeline and Stage, complete with layers. When you create a symbol you choose the symbol type, depending on how you want to use the symbol in your document.

- Use graphic symbols for static images and to create reusable pieces of animation that are tied to the main Timeline. Graphic symbols operate in sync with the main Timeline. Interactive controls and sounds won't work in a graphic symbol's animation sequence.
- Use button symbols to create interactive buttons that respond to mouse clicks, rollovers, or other actions. You define the graphics associated with various button states, and then assign actions to a button instance. For more information, see Chapter 5, "Handling Events" in *Using ActionScript in Flash*.
- Use movie clip symbols to create reusable pieces of animation. Movie clips have their own multiframe Timeline that is independent from the main Timeline—think of them as nested inside a main Timeline that can contain interactive controls, sounds, and even other movie clip instances. You can also place movie clip instances inside the Timeline of a button symbol to create animated buttons.

- Use font symbols to export a font and use it in other Flash documents. See "Creating font symbols" on page 116.

  Flash provides built-in *components*, movie clips with defined parameters, that you can use to add user interface elements, such as buttons, check boxes, or scroll bars, to your documents. For more information, see Introduction, "Getting Started with Components,"in *Using Components*.

Note: To preview interactivity and animation in movie clip symbols in the Flash authoring environment, you must select Control > Enable Live Preview.

About controlling instances and symbols with ActionScript

You can use ActionScript to control movie clip and button instances. The movie clip or button instance must have a unique instance name to be used with ActionScript. For information on assigning a name to an instance, see "Creating instances" on page 58. You can also use ActionScript to control movie clip or button symbols. For more information, see Chapter 5, "Handling Events" in *Using ActionScript in Flash*.

Creating symbols

You can create a symbol from selected objects on the Stage, or you can create an empty symbol and make or import the content in symbol-editing mode. You can also create font symbols in Flash. See "Creating font symbols" on page 116. Symbols can have all the functionality that you can create with Flash, including animation.

By using symbols that contain animation, you can create Flash applications with a lot of movement while minimizing file size. Consider creating animation in a symbol when there is a repetitive or cyclic action—the up-and-down motion of a bird's wings, for example.

You can also add symbols to your document by using shared library assets during authoring or at runtime. See "Using shared library assets" on page 69.

To convert selected elements to a symbol:

1. Select an element or several elements on the Stage. Then do one of the following:
   - Select Modify > Convert to Symbol.
   - Drag the selection to the Library panel.
   - Right-click (Windows) or Control-click (Macintosh) and select Convert to Symbol from the context menu.
2. In the Convert to Symbol dialog box, type the name of the symbol and select the behavior—Graphic, Button, or Movie Clip. See "Types of symbols" on page 54.
3. Click in the registration grid to position the registration point for the symbol.
4. Click OK.

   Flash adds the symbol to the library. The selection on the Stage becomes an instance of the symbol. You cannot edit an instance directly on the Stage—you must open it in symbol-editing mode. You can also change the registration point for a symbol. See "Editing symbols" on page 61.

To create a new empty symbol:

1. Make sure that nothing is selected on the Stage. Then do one of the following:
   - Select Modify > New Symbol.
   - Click the New Symbol button at the lower left of the Library panel.
   - Select New Symbol from the Library options menu in the upper right corner of the Library panel.
2. In the Create New Symbol dialog box, type the name of the symbol and select the behavior—Graphic, Button, or Movie Clip. See "Types of symbols" on page 54.
3. Click OK.

   Flash adds the symbol to the library and switches to symbol-editing mode. In symbol-editing mode, the name of the symbol appears above the upper left corner of the Stage, and a cross hair indicates the symbol's registration point.
4. To create the symbol content, use the Timeline, draw with the drawing tools, import media, or create instances of other symbols.
5. When you have finished creating the symbol content, do one of the following to return to document-editing mode:
   - Click the Back button at the left of the Edit bar above the Stage.
   - Select Edit > Edit Document.
   - Click the scene name in the Edit bar above the Stage.

   When you create a new symbol, the registration point is placed at the center of the window in symbol-editing mode. You can place the symbol contents in the window in relation to the registration point. You can also move the symbol contents in relation to the registration point when you edit a symbol, in order to change the registration point. See "Editing symbols" on page 61.

Converting animation on the Stage into a movie clip

If you've created an animated sequence on the Stage and want to reuse it elsewhere in your document, or if you want to manipulate it as an instance, you can select it and save it as a movie clip symbol.

To convert animation on the Stage into a movie clip:

1. On the main Timeline, select every frame in every layer of the animation on the Stage that you want to use. For information on selecting frames, see "Using the Timeline" in *Getting Started with Flash*.
2. Do one of the following to copy the frames:
   - Right-click (Windows) or Control-click (Macintosh) any selected frame and select Copy Frames from the context menu. Select Cut if you want to delete the sequence after converting it to a movie clip.
   - Select Edit > Timeline > Copy Frames. Select Cut Frames if you want to delete the sequence after converting it to a movie clip.

3. Deselect your selection and make sure nothing on the Stage is selected. Select Modify > New Symbol.
4. In the Create New Symbol dialog box, name the symbol. For Behavior, select Movie Clip, then click OK.

   Flash opens a new symbol for editing in symbol-editing mode.
5. On the Timeline, click Frame 1 on Layer 1, and select Edit > Timeline > Paste Frames.

   This pastes the frames (and any layers and layer names) you copied from the main Timeline to the Timeline of this movie clip symbol. Any animation, buttons, or interactivity from the frames you copied now becomes an independent animation (a movie clip symbol) that you can reuse throughout your document.
6. When you have finished creating the symbol content, do one of the following to return to document-editing mode:
   - Click the Back button at the left of the Edit bar above the Stage.
   - Select Edit > Edit Document.
   - Click the scene name in the Edit bar above the Stage.

Duplicating symbols

Duplicating a symbol lets you use an existing symbol as a starting point for creating a new symbol.

You can also use instances to create versions of the symbol with different appearances. See "Creating instances" on page 58.

To duplicate a symbol using the Library panel:

1. Select a symbol in the Library panel.
2. Do one of the following to duplicate the symbol:
   - Right-click (Windows) or Control-click (Macintosh) and select Duplicate from the context menu.
   - Select Duplicate from the Library options menu.

To duplicate a symbol by selecting an instance:

1. Select an instance of the symbol on the Stage.
2. Select Modify > Symbol > Duplicate Symbol.

   The symbol is duplicated and the instance is replaced with an instance of the duplicate symbol.

Creating instances

After you create a symbol, you can create instances of that symbol wherever you like throughout your document, including inside other symbols. When you modify the symbol, Flash updates all instances of the symbol.

Flash gives movie clip and button instances default instance names when you create them. From the Property inspector, you can apply custom names to instances. You use the instance name to refer to an instance in ActionScript. You must give each instance a unique name in order to control it with ActionScript. For more information, see Chapter 5, "Handling Events," in *Using ActionScript in Flash*.

To create a new instance of a symbol:

1. Select a layer in the Timeline.

   Flash can place instances only in keyframes, always on the current layer. If you don't select a keyframe, Flash adds the instance to the first keyframe to the left of the current frame.

   Note: A keyframe is a frame in which you define a change in the animation. For more information, see "Working with frames in the Timeline" in *Getting Started with Flash*.

2. Select Window > Library to open the library.
3. Drag the symbol from the library to the Stage.
4. If you created an instance of a graphic symbol, select Insert > Timeline > Frame to add the number of frames that will contain the graphic symbol.

To apply a custom name to an instance:

1. Select the instance on the Stage.
2. Select Window > Properties if the Property inspector is not visible.
3. Enter a name in the Instance Name text box on the left side of the Property inspector (below the Symbol Behavior pop-up list).

After creating an instance of a symbol, you can use the Property inspector to specify color effects, assign actions, set the graphic display mode, or change the behavior of the instance. The behavior of the instance is the same as the symbol behavior, unless you specify otherwise. Any changes you make affect only the instance and not the symbol. See "Changing instance properties" on page 62.

Creating buttons

Buttons are actually four-frame interactive movie clips. When you select the button behavior for a symbol, Flash creates a Timeline with four frames. The first three frames display the button's three possible states; the fourth frame defines the active area of the button. The Timeline doesn't actually play; it simply reacts to pointer movement and actions by jumping to the appropriate frame.

To make a button interactive, you place an instance of the button symbol on the Stage and assign actions to the instance. You must assign the actions to the instance of the button in the document, not to frames in the button's Timeline.

Each frame in the Timeline of a button symbol has a specific function:

- The first frame is the Up state, representing the button whenever the pointer is not over the button.
- The second frame is the Over state, representing the button's appearance when the pointer is over the button.
- The third frame is the Down state, representing the button's appearance as it is clicked.
- The fourth frame is the Hit state, defining the area that responds to the mouse click. This area is invisible in the SWF file.

You can also create a button using a movie clip symbol or a button component. There are advantages to using each type of button, depending on your needs. Creating a button using a movie clip enables you to add more frames to the button or add more complex animation. However, movie clip buttons have a larger file size than button symbols. Using a button component allows you to bind the button to other components, to share and display data in an application. Button components also include prebuilt features, such as accessibility support, and can be customized. Button components include the PushButton and RadioButton. For more information, see "Button component" in *Using Components*.

For a lesson on creating buttons with ActionScript, select Help > How Do I > Quick Tasks > Write Scripts with ActionScript.

To create a button:

1. Select Edit > Deselect All to ensure that nothing is selected on the Stage.
2. Select Insert > New Symbol, or press Control+F8 (Windows) or Command+F8 (Macintosh).

   To create the button, you convert the button frames to keyframes.
3. In the Create New Symbol dialog box, enter a name for the new button symbol, and for Behavior select Button.

   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.
4. 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 a graphic or movie clip symbol in a button, but you cannot use another button in a button. Use a movie clip symbol if you want the button to be animated.
5. Click the second frame, labeled Over, and select Timeline > Keyframe.

   Flash inserts a keyframe that duplicates the contents of the Up frame.
6. Change the button image for the Over state.

7. Repeat steps 5 and 6 for the Down frame and the Hit frame.

   The Hit frame is not visible on the Stage, but it defines the area of the button that responds when clicked. Make sure that the graphic for the Hit frame 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 image for the Up state is used as the Hit frame.

   You can create a disjoint rollover, in which moving the pointer over a button causes another graphic on the Stage to change. To do this, you place the Hit frame in a different location than the other button frames.

8. To assign a sound to a state of the button, select that state's frame in the Timeline, select Window > Properties, and then select a sound from the Sound menu in the Property inspector. For more information, see "Adding sounds to buttons" on page 204.

9. When you finish, select Edit > Edit Document. Drag the button symbol from the Library panel to create an instance of it in the document.

Enabling, editing, and testing buttons

By default, Flash keeps buttons disabled as you create them, to make it easier to select and work with them. When a button is disabled, clicking the button selects it. When a button is enabled, it responds to the mouse events that you've specified as if the SWF file were playing. You can still select enabled buttons, however. In general, it is best to disable buttons as you work, and enable buttons to quickly test their behavior.

To enable and disable buttons:

- Select Control > Enable Simple Buttons. A check mark appears next to the command to indicate buttons are enabled. Select the command again to disable buttons.

  Any buttons on the Stage now respond. As you move the pointer over a button, Flash displays the Over frame; when you click within the button's active area, Flash displays the Down frame.

To select an enabled button:

- Use the Selection tool to drag a selection rectangle around the button.

To move or edit an enabled button:

1. Select the button, as described above.
2. Do one of the following:
   - Use the arrow keys to move the button.
   - If the Property inspector is not visible, select Window > Properties to edit the button in the Property inspector, or Alt-double-click (Windows) or Option-double-click the button (Macintosh).

To test a button, do one of the following:

- Select Control > Enable Simple Buttons. Move the pointer over the enabled button to test it.
- Select the button in the Library panel and click the Play button in the Library preview window.
- Select Control > Test Scene or Control > Test Movie.

Movie clips in buttons are not visible in the Flash authoring environment. See "Enabling, editing, and testing buttons" on page 60.

Editing symbols

When you edit a symbol, Flash updates all the instances of that symbol in your document. Flash provides three ways for you to edit symbols. You can edit the symbol in context with the other objects on the Stage using the Edit in Place command. Other objects are dimmed to distinguish them from the symbol you are editing. The name of the symbol you are editing is displayed in an Edit bar at the top of the Stage, to the right of the current scene name.

You can also edit a symbol in a separate window, using the Edit in New Window command. Editing a symbol in a separate window lets you see the symbol and the main Timeline at the same time. The name of the symbol you are editing is displayed in the Edit bar at the top of the Stage.

You edit the symbol by changing the window from the Stage view to a view of only the symbol, using symbol-editing mode. The name of the symbol you are editing is displayed in the Edit bar at the top of the Stage, to the right of the current scene name.

When you edit a symbol, Flash updates all instances of the symbol throughout the document to reflect your edits. While editing a symbol, you can use any of the drawing tools, import media, or create instances of other symbols.

You can change the registration point of a symbol (the point identified by the coordinates 0, 0) using any symbol-editing method.

To edit a symbol in place:

1. Do one of the following:
   - Double-click an instance of the symbol on the Stage.
   - Select an instance of the symbol on the Stage and right-click (Windows) or Control-click (Macintosh), and select Edit in Place from the context menu.
   - Select an instance of the symbol on the Stage and select Edit > Edit in Place.
2. Edit the symbol as needed.
3. To change the registration point, drag the symbol on the Stage. A cross hair indicates the location of the registration point.
4. To exit edit-in-place mode and return to document-editing mode, do one of the following:
   - Click the Back button at the left of the Edit bar at the top of the Stage.
   - Select the current scene name from the Scene pop-up menu in the Edit bar at the top of the Stage.
   - Select Edit > Edit Document.

To edit a symbol in a new window:

1. Select an instance of the symbol on the Stage and right-click (Windows) or Control-click (Macintosh), and select Edit in New Window from the context menu.
2. Edit the symbol as needed.
3. To change the registration point, drag the symbol on the Stage. A cross hair indicates the location of the registration point.
4. Click the Close box in the upper right corner (Windows) or upper left corner (Macintosh) to close the new window, and click in the main document window to return to editing the main document.

To edit a symbol in symbol-editing mode:

1. Do one of the following to select the symbol:
   - Double-click the symbol's icon in the Library panel.
   - Select an instance of the symbol on the Stage and right-click (Windows) or Control-click (Macintosh) and select Edit from the context menu.
   - Select an instance of the symbol on the Stage and select Edit > Edit Symbols.
   - Select the symbol in the Library panel and select Edit from the Library options menu, or right-click (Windows) or Control-click (Macintosh) the symbol in the Library panel and select Edit from the context menu.
2. Edit the symbol as needed on the Stage.
3. To change the registration point, drag the symbol on the Stage. A cross hair indicates the location of the registration point.
4. To exit symbol-editing mode and return to editing the document, do one of the following:
   - Click the Back button at the left of the Edit bar at the top of the Stage.
   - Select Edit > Edit Document.
   - Click the scene name in the Edit bar at the top of the Stage.

Changing instance properties

Each symbol instance has its own properties that are separate from the symbol. You can change the tint, transparency, and brightness of an instance; redefine how the instance behaves (for example, change a graphic to a movie clip); and specify how animation plays inside a graphic instance. You can also skew, rotate, or scale an instance without affecting the symbol.

In addition, you can name a movie clip or button instance so that you can use ActionScript to change its properties. For more information, see Chapter 7, "Using the Built-In Classes," in *Using ActionScript in Flash*. To edit instance properties, you use the Property inspector (Windows > Properties).

The properties of an instance are saved with it. If you edit a symbol or relink an instance to a different symbol, any instance properties you've changed still apply to the instance.

Changing the color and transparency of an instance

Each instance of a symbol can have its own color effect. To set color and transparency options for instances, you use the Property inspector. Settings in the Property inspector also affect bitmaps placed within symbols.

When you change the color and transparency for an instance in a specific frame, Flash makes the change as soon as it displays that frame. To make gradual color changes, you must apply a motion tween. When tweening color, you enter different effect settings in starting and ending keyframes of an instance, and then tween the settings to make the instance's colors shift over time. See "Tweening instances, groups, and type" on page 165.

Note: If you apply a color effect to a movie clip symbol that has multiple frames, Flash applies the effect to every frame in the movie clip symbol.

To change the color and transparency of an instance:

1. Select the instance on the Stage and select Window > Properties.
2. In the Property inspector, select one of the following options from the Color pop-up menu:

   Brightness adjusts the relative lightness or darkness of the image, measured on a scale from black (–100%) to white (100%). Click the triangle and drag the slider or enter a value in the text box to adjust brightness.

   Tint colors the instance with the same hue. Use the Tint slider in the Property inspector to set the tint percentage, from transparent (0%) to completely saturated (100%). Click the triangle and drag the slider or enter a value in the text box to adjust tint. To select a color, enter red, green, and blue values in the respective text boxes, or click the color box and select a color from the pop-up window or click the Color Picker button.

   Alpha adjusts the transparency of the instance, from transparent (0%) to completely saturated (100%). To adjust the alpha value, click the triangle and drag the slider or enter a value in the text box.

   Advanced separately adjusts the red, green, blue, and transparency values of an instance. This is most useful when you want to create and animate subtle color effects on objects such as bitmaps. The controls on the left let you reduce the color or transparency values by a specified percentage. The controls on the right let you reduce or increase the color or transparency values by a constant value.

   The current red, green, blue, and alpha values are multiplied by the percentage values, and then added to the constant values in the right column, producing the new color values. For example, if the current red value is 100, setting the left slider to 50% and the right slider to 100 produces a new red value of 150 ([100 x .5] + 100 = 150).

   Note: The Advanced settings in the Effect panel implement the function (a * y+ b)= x where a is the percentage specified in the left set of text boxes, y is the color of the original bitmap, b is the value specified in the right set of text boxes, and x is the resulting effect (between 0 and 255 for RGB, and 0 and 100 for alpha transparency).

You can also change the color of an instance using the ActionScript Color object. For detailed information on the Color object, see "Color class" in *Flash ActionScript Language Reference.*

Swapping one instance for another

You can assign a different symbol to an instance to display a different instance on the Stage and preserve all the original instance properties, such as color effects or button actions.

For example, suppose you're creating a cartoon with a rat symbol for your character, but decide to change the character to a cat. You could replace the rat symbol with the cat symbol and have the updated character appear in roughly the same location in all your frames.

To assign a different symbol to an instance:

1. Select the instance on the Stage and select Window > Properties.
2. Click the Swap button in the Property inspector.
3. In the Swap Symbol dialog box, select a symbol to replace the one currently assigned to the instance. To duplicate a selected symbol, click the Duplicate Symbol button at the bottom of the dialog box.

   Duplicating lets you base a new symbol on an existing one in the library and minimizes copying if you're making several symbols that differ just slightly.
4. Click OK.

To replace all instances of a symbol:

1. Drag a symbol with the same name as the one you are replacing into the Library panel.
2. In the Resolve Library Item Conflict dialog box, click Replace. For more information, see "Resolving conflicts between library assets" on page 72.

Changing an instance's type

You can change an instance's type to redefine its behavior in a Flash application. For example, if a graphic instance contains animation that you want to play independently of the main Timeline, you could redefine the graphic instance as a movie clip instance.

To change an instance's type:

1. Select the instance on the Stage and select Window > Properties.
2. Select Graphic, Button, or Movie Clip from the pop-up menu in the upper left corner of the Property inspector.

Setting the animation for graphic instances

You can determine how animation sequences inside a graphic instance play in your Flash application by setting options in the Property inspector.

An animated graphic symbol is tied to the Timeline of the document in which the symbol is placed. In contrast, a movie clip symbol has its own independent Timeline. Animated graphic symbols, because they use the same Timeline as the main document, display their animation in document-editing mode. Movie clip symbols appear as static objects on the Stage and do not appear as animations in the Flash editing environment.

To set the animation of a graphic instance:

1. Select a graphic instance on the Stage and select Window > Properties.
2. In the Property inspector, select an animation option from the pop-up menu below the instance name:

   Loop loops all the animation sequences contained in the current instance for as many frames as the instance occupies.

   Play Once plays the animation sequence beginning from the frame you specify to the end of the animation and then stops.

   Single Frame displays one frame of the animation sequence. Specify which frame to display.

Controlling instances with behaviors

You can use behaviors to control movie clip and graphic instances in a document without writing ActionScript. Behaviors are prewritten ActionScript scripts that let you add the power, control, and flexibility of ActionScript coding to your document without having to create the ActionScript code yourself.

You can use behaviors with an instance to arrange it in the stacking order on a frame, as well as to load or unload, play, stop, duplicate, or drag a movie clip, or to link to a URL.

In addition, you can use behaviors to load an external graphic or an animated mask into a movie clip.

To control a movie clip with a behavior, you use the Behaviors panel to apply the behavior to a triggering object, such as a button. You specify the event that triggers the behavior (such as releasing the button), select a target object (the movie clip instance) that is affected by the behavior, and when necessary, specify settings for the behavior parameters, such as a frame number or label.

The behaviors in the following table are packaged with Flash MX 2004 and Flash MX Professional 2004. For more information on embedded video behaviors, see "Controlling video playback using behaviors" on page 191. For more information on controlling sounds with behaviors, see "Controlling sound playback using behaviors" on page 206.

Behavior	Purpose	Select/input
Load Graphic	Loads an external JPEG file into a movie clip or screen.	Path and filename of JPEG file. Instance name of movie clip or screen receiving the graphic.
Load External Movie Clip	Loads an external SWF file into a target movie clip or screen.	URL of external SWF file. Instance name of movie clip or screen receiving the SWF file.
Duplicate Movieclip	Duplicates a movie clip or screen	Instance name of movie clip to duplicate. X-offset and Y-offset of pixels from original to copy.
GotoAndPlay at frame or label	Plays a movie clip from a particular frame.	Instance name of target clip to play. Frame number or label to play.

Behavior	Purpose	Select/input
GotoAndStop at frame or label	Stops a movie clip, optionally moving the playhead to a particular frame.	Instance name of target clip to stop. Frame number or label to stop.
Bring to Front	Brings target movie clip or screen to the top of the stacking order.	Instance name of movie clip or screen.
Bring Forward	Brings target movie clip or screen one position higher in the stacking order.	Instance name of movie clip or screen.
Send to Back	Sends the target movie clip to the bottom of the stacking order.	Instance name of movie clip or screen.
Send Backward	Sends the target movie clip or screen one position lower in the stacking order.	Instance name of movie clip or screen.
Start Dragging movieclip	Starts dragging a movie clip.	Instance name of movie clip or screen.
Stop Dragging movieclip	Stops the current drag.	

To add and configure a behavior:

1. Select the object, such as a button, that will trigger the behavior.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button and select the desired behavior from the Movieclip submenu.
3. In the dialog box that appears, select the movie clip that you want to control with the behavior.
4. Select a relative or absolute path. For more information, see "Absolute paths" on page 24 and "Relative paths" on page 25.
5. If required, select or input settings for the behavior parameters and click OK.

   Default settings for the behavior appear in the Behaviors panel.
6. Under Event, click On Release (the default event) and select a mouse event from the menu. If you want to use the On Release event, leave the option unchanged.

Breaking apart instances

To break the link between an instance and a symbol and make the instance into a collection of ungrouped shapes and lines, you "break apart" the instance. This is useful for changing the instance substantially without affecting any other instance. If you modify the source symbol after breaking apart the instance, the instance is not updated with the changes.

To break apart an instance of a symbol:

1. Select the instance on the Stage.
2. Select Modify > Break Apart.

   This breaks the instance into its component graphic elements.
3. Use the painting and drawing tools to modify these elements as desired.

Getting information about instances on the Stage

As you create a Flash application, it can be difficult to identify a particular instance of a symbol on the Stage, particularly if you are working with multiple instances of the same symbol. You can identify instances using the Property inspector, the Info panel, or the Movie Explorer.

The Property inspector and Info panel display the symbol name of the selected instance and an icon that indicates its type—graphic, button, or movie clip. In addition, you can view the following information:

- In the Property inspector, you can view the instance's behavior and settings—for all instance types, color effect settings, location, and size; for graphics, the loop mode and first frame that contains the graphic; for buttons, the instance name (if assigned) and tracking option; for movie clips, the instance name (if assigned). For location, the Property inspector displays the *x* and *y* coordinates of either the symbol's registration point or the symbol's upper left corner, depending on which option is selected in the Info panel.
- In the Info panel, you can view the instance's size and location; the location of its registration point; its red (R), green (G), blue (B), and alpha (A) values (if the instance has a solid fill); and the location of the pointer. The Info panel also displays the *x* and *y* coordinates of either the symbol's registration point or the symbol's upper left corner, depending on which option is selected. To display the coordinates of the registration point, click the center square in the Coordinate grid in the Info panel. To display the coordinates of the upper left corner, click the upper left square in the Coordinate grid.
- In the Movie Explorer, you can view the contents of the current document, including instances and symbols. See "Using the Movie Explorer" on page 28.

In addition, in the Actions panel, you can view any actions assigned to a button or movie clip.

To get information about an instance on the Stage:

1. Select the instance on the Stage.
2. Display the Property inspector or panel you want to use:
   - To display the Property inspector, select Window > Properties.
   - To display the Info panel, select Window > Design Panels > Info.
   - To display the Movie Explorer, select Window > Other Panels > Movie Explorer. For more information on the Movie Explorer, see "Using the Movie Explorer" on page 28.
   - To display the Actions panel, select Window > Development Panels > Actions.

To view the symbol definition for the selected symbol in the Movie Explorer:

1. Click the Show Buttons, Movie Clips, and Graphics button at the top of the Movie Explorer.
2. Right-click (Windows) or Control-click (Macintosh) and select Show Symbol Instances and Go to Symbol Definition from the context menu; or select these options from the pop-up menu in the upper right corner of the Movie Explorer.

To jump to the scene containing instances of a selected symbol:

1. Display the symbol definitions as described in the previous procedure.
2. Right-click (Windows) or Control-click (Macintosh) and select Show Movie Elements and Go to Symbol Definition from the context menu; or select these options from the pop-up menu in the upper right corner of the Movie Explorer.

Copying library assets between documents

You can copy library assets from a source document into a destination document in a variety of ways: by copying and pasting the asset, by dragging and dropping the asset, or by opening the library of the source document in the destination document and dragging the source document assets into the destination document.

You can also share symbols between documents as shared library assets during authoring or at runtime. See "Using shared library assets" on page 69.

If you attempt to copy assets that have the same name as existing assets in the destination document, the Resolve Library Conflicts dialog box lets you choose whether to overwrite the existing assets or to preserve the existing assets and add the new assets with modified names. See "Resolving conflicts between library assets" on page 72. You can organize library assets in folders to minimize name conflicts when copying assets between documents. See "Working with folders in the Library panel" on page 19.

To copy a library asset by copying and pasting:

1. Select the asset on the Stage in the source document.
2. Select Edit > Copy.
3. Make the destination document the active document.
4. Place the pointer on the Stage and select Edit > Paste in Center to paste the asset in the center of the visible work area. Select Edit > Paste in Place to place the asset in the same location as in the source document.

To copy a library asset by dragging:

1. With the destination document open in Flash, select the asset in the Library panel in the source document.
2. Drag the asset into the Library panel in the destination document.

To copy a library asset by opening the source document **library in the destination document:**

1. With the destination document active in Flash, select File > Import > Open External Library.
2. Select the source document in the Open As Library dialog box and click Open.
3. Drag an asset from the source document library onto the Stage or into the library of the destination document.

Using shared library assets

Shared library assets let you use assets from a source document in multiple destination documents.You can share library assets in two different ways:

- For runtime shared assets, assets from a source document are linked as external files in a destination document. Runtime assets are loaded into the destination document during document playback—that is, at runtime. The source document containing the shared asset does not need to be available on your local network when you author the destination document. However, the source document must be posted to a URL in order for the shared asset to be available to the destination document at runtime.
- For shared assets during authoring, you can update or replace any symbol in a document you are authoring with any other symbol available on your local network. You can update the symbol in the destination document as you author the document. The symbol in the destination document retains its original name and properties, but its contents are updated or replaced with those of the symbol you select.

Using shared library assets can optimize your workflow and document asset management in numerous ways. For example, you can use shared library assets to share a font symbol across multiple sites, provide a single source for elements in animations used across multiple scenes or document, or create a central resource library to use for tracking and controlling revisions.

Working with runtime shared assets

Using runtime shared library assets involves two procedures: First, the author of the source document defines a shared asset in the source document and enters an identifier string for the asset and a URL where the source document will be posted.

Second, the author of the destination document defines a shared asset in the destination document and enters an identifier string and URL identical to those used for the shared asset in the source document. Alternatively, the destination document author can drag the shared assets from the posted source document into the destination document library.

In either scenario, the source document must be posted to the specified URL in order for the shared assets to be available for the destination document.

Defining runtime shared assets in a source document

You use the Symbol Properties dialog box or the Linkage Properties dialog box to define sharing properties for an asset in a source document, to make the asset accessible for linking to destination documents.

To define a runtime shared asset in a source document:

1. With the source document open, select Window > Library to display the Library panel.
2. Do one of the following:
   - Select a movie clip, button, or graphic symbol in the Library panel and select Properties from the Library options menu. Click the Advanced button to expand the Properties dialog box.
   - Select a font symbol, sound, or bitmap and select Linkage from the Library options menu.
3. For Linkage, select Export for Runtime Sharing to make the asset available for linking to the destination document.
4. Enter an identifier for the symbol in the Identifier text field. Do not include spaces. This is the name Flash uses to identify the asset when linking to the destination document.

   Note: Flash also uses the linkage identifier to identify a movie clip or button that is used as an object in ActionScript. See Chapter 8, "Working with Movie Clips," in *Using ActionScript in Flash*.
5. Enter the URL where the SWF file containing the shared asset will be posted.
6. Click OK.

   When you publish the SWF file, you must post the SWF file to the URL specified in step 5, so that the shared assets will be available to destination documents.

Linking to runtime shared assets from a destination document

You use the Symbol Properties dialog box or the Linkage Properties dialog box to define sharing properties for an asset in a destination document so that you can link the asset to a shared asset in a source document. If the source document is posted to a URL, you can also link a shared asset to a destination document by dragging the asset from the source document to the destination document.

To embed the symbol in the destination document, you can turn off sharing for a shared asset in the destination document.

To link a shared asset to a destination document by entering the identifier and URL:

1. In the destination document, select Window > Library to display the Library panel.
2. Do one of the following:
   - Select a movie clip, button, or graphic symbol in the Library panel and select Properties from the Library options menu. Click the Advanced button to expand the Properties dialog box.
   - Select a font symbol and select Linkage from the Library options menu.
3. For Linkage, select Import for Runtime Sharing to link to the asset in the source document.
4. Enter an identifier for the symbol in the Identifier text field that is identical to the identifier used for the symbol in the source document. Do not include spaces.
5. Enter the URL where the SWF source file containing the shared asset is posted.
6. Click OK.

To link a shared asset to a destination document by dragging:

1. In the destination document, do one of the following:
   - Select File > Open.
   - Select File > Import > Open External Library.
2. In the Open or Open as Library dialog box, select the source document and click Open.
3. Drag the shared asset from the source document Library panel into the Library panel or onto the Stage in the destination document.

To turn off linkage for a symbol in a destination document:

1. In the destination document, select the linked symbol in the Library panel and do one of the following:
   - If the asset is a movie clip, button, or graphic symbol, select Properties from the Library options menu.
   - If the asset is a font symbol, select Linkage from the Library options menu.
2. In the Symbol Properties dialog box or the Linkage Properties dialog box, deselect Import for Runtime Sharing.
3. Click OK.

Updating or replacing symbols using sharing during authoring

You can update or replace a movie clip, button, or graphic symbol in a document with any other symbol in a FLA file accessible on your local network. The original name and properties of the symbol in the destination document are preserved, but the contents of the symbol are replaced with the contents of the symbol you select. Any assets that the selected symbol uses are also copied into the destination document.

To update or replace a symbol:

1. With the document open, select a movie clip, button, or graphic symbol and select Properties from the Library options menu.
2. If the Symbol Properties dialog box is in basic mode, click Advanced to display the Linkage and Source panels. If the Linkage and Source panel are open, go to Step 3.
3. To select a new FLA file, under Source in the Symbol Properties dialog box, click Browse.
4. In the Open dialog box, navigate to a FLA file containing the symbol that will be used to update or replace the selected symbol in the Library panel, and click Open.
5. To select a new symbol in the FLA file, under Source, click Symbol.
6. Navigate to a symbol and click Open.
7. In the Symbol Properties dialog box, under Source, select Always Update Before Publishing to automatically update the asset if a new version is found at the specified source location.
8. Click OK to close the Symbol Properties or Linkage Properties dialog box.

Resolving conflicts between library assets

If you import or copy a library asset into a document that already contains a different asset of the same name, you can choose whether to replace the existing item with the new item. This option is available with all the methods for importing or copying library assets, including the following:

- Copying and pasting an asset from a source document
- Dragging an asset from a source document or a source document library
- Importing an asset
- Adding a shared library asset from a source document
- Using a component from the components panel

The Resolve Library Items dialog box appears when you attempt to place items that conflict with existing items in a document. A conflict exists when you copy an item from a source document that already exists in the destination document and the items have different modification dates. You can avoid naming conflicts by organizing your assets inside folders in your document's library. The dialog box also appears when you paste a symbol or component into your document's Stage and you already have a copy of the symbol or component that has a different modification date from the one you're pasting.

If you choose not to replace the existing items, Flash attempts to use the existing item instead of the conflicting item that you are pasting. For example, if you copy a symbol named Symbol 1 and paste the copy into the Stage of a document that already contains a symbol named Symbol 1, Flash creates an instance of the existing Symbol 1.

If you choose to replace the existing items, Flash replaces the existing items (and all their instances) with the new items of the same name. If you cancel the Import or Copy operation, the operation is canceled for all items (not just those items that conflict in the destination document).

Only identical library item types may be replaced with each other. That is, you cannot replace a sound named Test with a bitmap named Test. In such cases, the new items are added to the library with the word Copy appended to the name.

Note: Replacing library items using this method is not undoable. Be sure to save a backup of your FLA file before performing complex paste operations that are resolved by replacing conflicting library items.

If the Resolve Library Conflict dialog box appears when you are importing or copying library assets into a document, you can resolve the naming conflict.

To resolve naming conflicts between library assets, do one of the following:

- Click Don't Replace Existing Items to preserve the existing assets in the destination document.
- Click Replace Existing Items to replace the existing assets and their instances with the new items of the same name.

CHAPTER 4
Working with Color

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 provide a variety of ways to apply, create, and modify colors. Using the default palette or a palette you create, you can choose colors to apply to the stroke or fill of an object you are about to create, or one already on the Stage. Applying a stroke color to a shape paints the outline of the shape with that color. Applying a fill color to a shape paints the interior space of the shape with that color.

When applying a stroke color to a shape, you can select any solid color, and you can select the style and weight of the stroke. For a shape's fill, you can apply a solid color, gradient, or bitmap. To apply a bitmap fill to a shape, you must import a bitmap into the current file. You can also create an outlined shape with no fill by using No Color as a fill, or you can create a filled shape with no outline by using No Color as an outline. And you can apply a solid color fill to text. See "Setting text attributes" on page 111.

You can modify stroke and fill attributes in a variety of ways using the Paint Bucket, Ink Bottle, Eyedropper, and Fill Transform tools, and the Lock Fill modifier for the Brush or Paint Bucket tools.

With the Color Mixer you can easily create and edit solid colors and gradient fills in RGB and HSB modes. You can import, export, delete, and otherwise modify the color palette for a file using the Color Swatches panel. You can select colors in hexadecimal mode in the Color Mixer, as well as in the Stroke and Fill pop-up windows in the Tools panel or Property inspector.

You can access the system color picker from the Stroke Color or Fill Color control in the Tools panel, the shape Property inspector, or the Color Mixer.

To access the system color picker:

- Alt-double-click (Windows) or Option-double-click (Macintosh) the Stroke Color or Fill Color control in the Tools panel, the shape Property inspector, or the Color Mixer.

This chapter contains the following sections:

Using the Stroke Color and Fill Color controls in the Tools panel

The Stroke Color and Fill Color controls in the Tools panel let you select a solid stroke color or a solid or gradient fill color, switch the stroke and fill colors, or select the default stroke and fill colors (black stroke and white fill). Oval and rectangle objects (shapes) can have both stroke and fill colors. Text objects and brush strokes can have only fill colors. Lines drawn with the Line, Pen, and Pencil tools can have only stroke colors.

The Tools panel Stroke Color and Fill Color controls set the painting attributes of new objects you create with the drawing and painting tools. To use these controls to change the painting attributes of existing objects, you must first select the objects on the Stage.

Note: Gradient swatches appear only in the Fill Color control.

To apply stroke and fill colors using the Tools panel controls, do one of the following:

- Click the triangle next to the Stroke or Fill color box and select a color swatch from the pop-up window. Gradients can be selected for the fill color only.
- Click the Color Picker button in the color pop-up window and select a color from the Color Picker.
- Type a color's hexadecimal value in the text box in the color pop-up window.
- Click the Default Fill and Stroke button in the Tools panel to return to the default color settings (white fill and black stroke).
- Click the No Color button in the color pop-up window to remove any stroke or fill.

  Note: The No Color button appears only when you are creating a new oval or rectangle. You can create a new object without a stroke or fill, but you cannot use the No Color button with an existing object. Instead, select the existing stroke or fill and delete it.

- Click the Swap Fill and Stroke button in the Tools panel to swap colors between the fill and the stroke.

Using the Stroke Color and Fill Color controls in the Property inspector

To change the stroke color, style, and weight for a selected object, you can use the Stroke Color controls in the Property inspector. For stroke style, you can choose from styles that are preloaded with Flash, or you can create a custom style.

To select a solid color fill, you can use the Fill Color control in the Property inspector.

To select a stroke color, style, and weight using the Property inspector:

1. Select an object or objects on the Stage (for symbols, first double-click to enter symbol-editing mode).
2. If the Property inspector is not visible, select Window > Properties.
3. To select a color, click the triangle next to the Stroke color box and do one of the following:
   - Select a color swatch from the palette.
   - Type a color's hexadecimal value in the text box.
4. To select a stroke style, click the triangle next to the Style pop-up menu and select an option from the menu. To create a custom style, select Custom from the Property inspector, then select options in the Stroke Style dialog box and click OK.

   Note: Selecting a stroke style other than Solid can increase file size.

5. To select a stroke weight, click the triangle next to the Weight pop-up menu and set the slider at the desired weight.

To apply a solid color fill using the Property inspector:

1. Select an object or objects on the Stage.
2. Select Window > Properties.
3. To select a color, click the triangle next to the Fill color box and do one of the following:
   - Select a color swatch from the palette.
   - Type a color's hexadecimal value in the text box.

Working with solid colors and gradient fills in the Color Mixer

To create and edit solid colors and gradient fills, you can use the Color Mixer. If an object is selected on the Stage, the color modifications you make in the Color Mixer are applied to the selection.

You can create any color using the Color Mixer. You can select colors in RGB or HSB, or you can expand the panel to use hexadecimal mode. You can also specify an alpha value to define the degree of transparency for a color. In addition, you can select a color from the existing color palette.

You can expand the Color Mixer to display a larger color space in place of the color bar, a split color swatch showing the current and previous colors, and a Brightness control to modify color brightness in all color modes.

To create or edit a solid color with the Color Mixer:

1. To apply the color to existing artwork, select an object or objects on the Stage.
2. Select Window > Design Panels > Color Mixer.
3. To select a color mode display, select RGB (the default setting) or HSB from the pop-up menu in the upper right corner of the Color Mixer.
4. Click the Stroke or Fill icon to specify which attribute is to be modified.

   Note: Be sure to click the icon, not the color box, or the color pop-up window will open.

5. If you selected the Fill icon in step 4, verify that Solid is selected in the Fill Style pop-up menu in the center of the Color Mixer.
6. Click the arrow in the lower right corner to expand the Color Mixer.
7. Do one of the following:
   - Click in the color space in the Color Mixer to select a color. Drag the Brightness control to adjust the brightness of the color.

     Note: To create colors other than black or white, make sure the Brightness control is not set to either extreme.

   - Enter values in the color value boxes: Red, Green, and Blue values for RGB display; Hue, Saturation, and Brightness values for HSB display; or hexadecimal values for hexadecimal display. Enter an Alpha value to specify the degree of transparency, from 0 for complete transparency to 100 for complete opacity.
   - Click the Default Stroke and Fill button to return to the default color settings, black and white (white fill and black stroke).
   - Click the Swap Stroke and Fill button to swap colors between the fill and the stroke.
   - Click the No Color button to apply no color to the fill or stroke.

     Note: You cannot apply a stroke or fill of No Color to an existing object. Instead, select the existing stroke or fill and delete it.

   - Click the Stroke or Fill color box and select a color from the pop-up window.
8. To add the color defined in step 7 to the color swatch list for the current document, select Add Swatch from the pop-up menu in the upper right corner of the Color Mixer.

To create or edit a gradient fill with the Color Mixer:

1. To apply a gradient fill to existing artwork, select an object or objects on the Stage.
2. If the Color Mixer is not visible, select Window > Design Panels > Color Mixer.
3. To select a color mode display, select RGB (the default setting) or HSB.
4. Select a gradient type from the Fill Style pop-up menu in the center of the Color Mixer:

   Linear Gradient creates a gradient that shades from the starting point to the end point in a straight line.

   Radial Gradient creates a gradient that shades from the starting point to the end point in a circular pattern.

   The gradient definition bar appears in place of the color bar in the Color Mixer, with pointers below the bar indicating each color in the gradient.
5. Click the arrow in the lower right corner to expand the Color Mixer.
6. To change a color in the gradient, click one of the pointers below the gradient definition bar and click in the color space that appears directly below the gradient bar in the expanded Color Mixer. Drag the Brightness control to adjust the lightness of the color.
7. To add a pointer to the gradient, click on or below the gradient definition bar. Select a color for the new pointer as described in step 6.

8. To reposition a pointer on the gradient, drag the pointer along the gradient definition bar. Drag a pointer down and off of the gradient definition bar to remove it.
9. To save the gradient, click the triangle in the upper right corner of the Color Mixer and select Add Swatch from the pop-up menu. The gradient is added to the Color Swatches panel for the current document.

Modifying strokes with the Ink Bottle tool

To change the stroke color, width, and style of lines or shape outlines, you can use the Ink Bottle tool. You can apply only solid colors, not gradients or bitmaps, to lines or shape outlines.

Using the Ink Bottle tool, rather than selecting individual lines, makes it easier to change the stroke attributes of multiple objects at one time.

To use the Ink Bottle tool:

1. Select the Ink Bottle tool from the Tools panel.
2. Select a stroke color as described in "Using the Stroke Color and Fill Color controls in the Tools panel" on page 76.
3. Select a stroke style and stroke width from the Property inspector. See "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.
4. Click an object on the Stage to apply the stroke modifications.

Applying solid, gradient, and bitmap fills with the Paint Bucket tool

The Paint Bucket tool fills enclosed areas with color. This tool lets you fill empty areas and change the color of already painted areas. You can paint with solid colors, gradient fills, and bitmap fills. You can use the Paint Bucket tool to fill areas that are not entirely enclosed, and you can have Flash close gaps in shape outlines as you use the Paint Bucket tool. See "Working with imported bitmaps" on page 138.

To use the Paint Bucket tool to fill an area:

1. Select the Paint Bucket tool from the Tools panel.
2. Select a fill color and style, as described in "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.
3. Click the Gap Size modifier and select a gap size option:
   - Select Don't Close Gaps if you want to close gaps manually before filling the shape. Closing gaps manually can be faster for complex drawings.
   - Select a Close option to have Flash fill a shape that has gaps.

   Note: If gaps are too large, you might have to close them manually.
4. Click the shape or enclosed area that you want to fill.

Transforming gradient and bitmap fills

You can transform a gradient or bitmap fill by adjusting the size, direction, or center of the fill. To transform a gradient or bitmap fill, you use the Fill Transform tool.

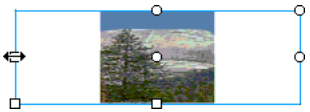
To adjust a gradient or bitmap fill with the Fill Transform tool:

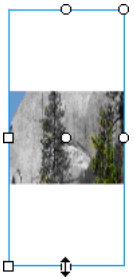
1. Select the Fill Transform tool.
2. Click an area filled with a gradient or bitmap fill.

   When you select a gradient or bitmap fill for editing, its center point appears, and its bounding box is displayed with editing handles. When the pointer is over any one of these handles, it changes to indicate the function of the handle.

   Press Shift to constrain the direction of a linear gradient fill to multiples of 45°.
3. Reshape the gradient or fill in any of the following ways:
   - To reposition the center point of the gradient or bitmap fill, drag the center point.

     
   - To change the width of the gradient or bitmap fill, drag the square handle on the side of the bounding box. (This option resizes only the fill, not the object containing the fill.)

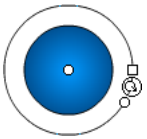
     
   - To change the height of the gradient or bitmap fill, drag the square handle at the bottom of the bounding box.

     
   - To rotate the gradient or bitmap fill, drag the circular rotation handle at the corner. You can also drag the lowest handle on the bounding circle of a circular gradient or fill.

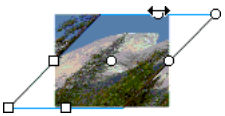
     

- To scale a linear gradient or a fill, drag the square handle at the center of the bounding box.

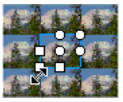
- To change the radius of a circular gradient, drag the middle circular handle on the bounding circle.



- To skew or slant a fill within a shape, drag one of the circular handles on the top or right side of the bounding box.



- To tile a bitmap inside a shape, scale the fill.



Note: To see all the handles when working with large fills or fills close to the edge of the Stage, select View › Work Area.

Copying strokes and fills with the Eyedropper tool

You can use the Eyedropper tool to copy fill and stroke attributes from one object and immediately apply them to another object. The Eyedropper tool also lets you sample the image in a bitmap to use as a fill. See "Breaking apart groups and objects" on page 155.

To use the Eyedropper tool to copy and apply stroke or fill attributes:

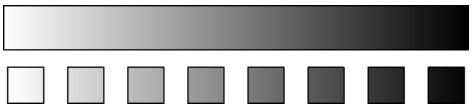
1. Select the Eyedropper tool and click the stroke or filled area whose attributes you want to apply to another stroke or filled area.

   When you click a stroke, the tool automatically changes to the Ink Bottle tool. When you click a filled area, the tool automatically changes to the Paint Bucket tool with the Lock Fill modifier turned on. See "Locking a gradient or bitmap to fill the Stage" on page 82.
2. Click another stroke or filled area to apply the new attributes.

Locking a gradient or bitmap to fill the Stage

You can lock a gradient or bitmap fill to make it appear that the fill extends over the entire Stage and that the objects painted with the fill are masks revealing the underlying gradient or bitmap. For information on applying a bitmap fill, see "Applying a bitmap fill" on page 140.

When you select the Lock Fill modifier with the Brush or Paint Bucket tool and paint with the tool, the bitmap or gradient fill extends across the objects you paint on the Stage.



Using the Lock Fill modifier creates the appearance of a single gradient or bitmap fill being applied to separate objects on the Stage.

To use a locked gradient fill:

1. Select the Brush or Paint Bucket tool and select a gradient or bitmap as a fill.
2. Select Linear Gradient or Radial Gradient from the Fill Style pop-up menu in the center of the Color Mixer and then select the Brush or Paint Bucket tool.
3. Click the Lock Fill modifier.
4. First paint the areas where you want to place the center of the fill, and then move to other areas.

To use a locked bitmap fill:

1. Select the bitmap you want to use.
2. Select Bitmap from the Fill Style pop-up menu in the center of the Color Mixer before selecting the Brush or Paint Bucket tool.
3. Select the Brush or Paint Bucket tool.
4. Click the Lock Fill modifier.
5. First paint the areas where you want to place the center of the fill, and then move to other areas.

Modifying color palettes

Each Flash file contains its own color palette, stored in the Flash document. Flash displays a file's palette as swatches in the Fill Color and Stroke Color controls and in the Color Swatches panel. The default color palette is the web-safe palette of 216 colors. You can add colors to the current color palette using the Color Mixer. See "Working with solid colors and gradient fills in the Color Mixer" on page 77.

To import, export, and modify a file's color palette, you use the Color Swatches panel. You can duplicate colors, remove colors from the palette, change the default palette, reload the web-safe palette if you have replaced it, or sort the palette according to hue.

You can import and export both solid and gradient color palettes between Flash files, as well as between Flash and other applications, such as Macromedia Fireworks and Adobe Photoshop.

Duplicating and removing colors

You can duplicate colors in the palette, delete individual colors, or clear all colors from the palette.

To duplicate a color or delete a color:

1. If the Color Swatches panel is not visible, select Window > Design Panels > Color Swatches.
2. Click the color that you want to duplicate or delete.
3. Select Duplicate Swatch or Delete Swatch from the pop-up menu in the upper right corner.

To clear all colors from the color palette:

- In the Color Swatches panel, select Clear Colors from the pop-up menu in the upper right corner. All colors are removed from the palette except black and white.

Using the default palette and the web-safe palette

You can save the current palette as the default palette, replace the current palette with the default palette defined for the file, or load the web-safe palette to replace the current palette.

To load or save the default palette:

- In the Color Swatches panel, select one of the following commands from the pop-up menu in the upper right corner:

  Load Default Colors replaces the current palette with the default palette.

  Save as Default saves the current color palette as the default palette. The new default palette is used when you create new files.

To load the web-safe 216-color palette:

- In the Color Swatches panel, select Web 216 from the pop-up menu in the upper right corner.

Sorting the palette

To make it easier to locate a color, you can sort colors in the palette by hue.

To sort colors in the palette:

- In the Color Swatches panel, select Sort by Color from the pop-up menu in the upper right corner.

Importing and exporting color palettes

To import and export both RGB colors and gradients between Flash files, you use Flash Color Set files (CLR files). You can import and export RGB color palettes using Color Table files (ACT files) that can be used with Macromedia Fireworks and Adobe Photoshop. You can also import color palettes, but not gradients, from GIF files. You cannot import or export gradients from ACT files.

To import a color palette:

1. In the Color Swatches panel, select one of the following commands from the pop-up menu in the upper right corner:
   - To append the imported colors to the current palette, select Add Colors.
   - To replace the current palette with the imported colors, select Replace Colors.
2. Navigate to the desired file and select it.
3. Click OK.

To export a color palette:

1. In the Color Swatches panel, select Save Colors from the pop-up menu in the upper right corner.
2. In the dialog box that appears, enter a name for the color palette.
3. For Save As Type (Windows) or Format (Macintosh), select Flash Color Set or Color Table. Click Save.

CHAPTER 5
Drawing

The drawing tools in Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 let you create and modify shapes for the artwork in your documents. For an interactive introduction to drawing in Flash, select Help > How Do I > Basic Flash > Draw in Flash.

Before you draw and paint in Flash, it is important to understand how Flash creates artwork, how drawing tools work, and how drawing, painting, and modifying shapes can affect other shapes on the same layer.

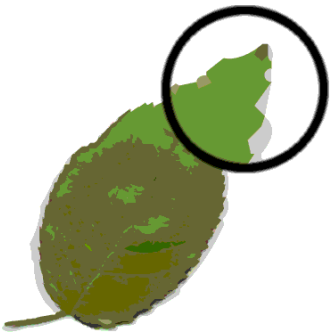
This chapter contains the following sections:

About vector and bitmap graphics

Computers display graphics in either vector or bitmap format. Understanding the difference between the two formats can help you work more efficiently. Using Flash, you can create and animate compact vector graphics. Flash also lets you import and manipulate vector and bitmap graphics that have been created in other applications.

Vector graphics

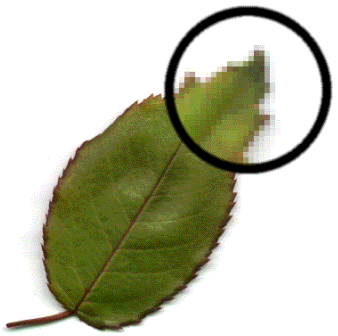
Vector graphics describe images using lines and curves, called *vectors,* that also include color and position properties. For example, the image of a leaf is described by points through which lines pass, creating the leaf's outline. The color of the leaf is determined by the color of the outline and the color of the area enclosed by the outline.



When you edit a vector graphic, you modify the properties of the lines and curves that describe its shape. You can move, resize, reshape, and change the color of a vector graphic without changing the quality of its appearance. Vector graphics are resolution-independent; that is, they can be displayed on output devices of varying resolutions without losing any quality.

Bitmap graphics

Bitmap graphics describe images using colored dots, called *pixels,* arranged in a grid. For example, the image of a leaf is described by the specific location and color value of each pixel in the grid, creating an image in much the same manner as a mosaic.



When you edit a bitmap graphic, you modify pixels rather than lines and curves. Bitmap graphics are resolution-dependent, because the data describing the image is fixed to a grid of a particular size. Editing a bitmap graphic can change the quality of its appearance. In particular, resizing a bitmap graphic can make the edges of the image ragged as pixels are redistributed within the grid. Displaying a bitmap graphic on an output device that has a lower resolution than the image itself also degrades its quality.

Flash drawing and painting tools

Flash provides various tools for drawing freeform or precise lines, shapes, and paths, and for painting filled objects.

- To draw freeform lines and shapes as if drawing with a real pencil, you use the Pencil tool. See "Drawing with the Pencil tool" on page 88.
- To draw precise paths as straight or curved lines, you use the Pen tool. See "Using the Pen tool" on page 90.
- To draw basic geometric shapes, you use the Line, Oval, and Rectangle tools. See "Drawing straight lines, ovals, and rectangles" on page 89.
- To draw polygons and stars, you use the PolyStar tool. See "Drawing polygons and stars" on page 90.
- To create brushlike strokes as if painting with a brush, you use the Brush tool. See "Painting with the Brush tool" on page 95.

When you use most Flash tools, the Property inspector changes to present the settings associated with that tool. For example, if you select the Text tool, the Property inspector displays text properties, making it easy to select the text attributes you want. For more information on the Property inspector, see "Using panels and the Property inspector" in *Getting Started with Flash*.

When you use a drawing or painting tool to create an object, the tool applies the current stroke and fill attributes to the object. To change the stroke and fill attributes of existing objects, you can use the Paint Bucket and Ink Bottle tools in the Tools panel or the Property inspector. See "Using the Stroke Color and Fill Color controls in the Tools panel" on page 76 or "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.

You can reshape lines and shape outlines in a variety of ways after you create them. Fills and strokes are treated as separate objects. You can select fills and strokes separately to move or modify them. See "Reshaping lines and shape outlines" on page 97.

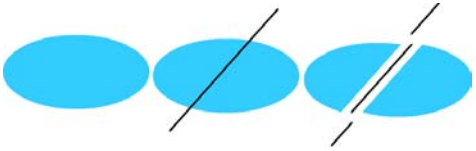
You can use snapping to automatically align elements with each other and with the drawing grid or guides. See "Snapping" on page 101 and"About the main toolbar and edit bar" in *Getting Started with Flash*.

You can customize the Tools panel to change the display of tools. See "Customizing the Tools panel" in *Getting Started with Flash*.

About overlapping shapes in Flash

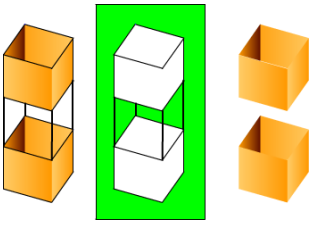
When you use the Pencil, Line, Oval, Rectangle, or Brush tool to draw a line across another line or painted shape, the overlapping lines are divided into segments at the intersection points. You can use the Selection tool to select, move, and reshape each segment individually.

Note: Overlapping lines that you create with the Pen tool do not divide into individual segments at intersection points, but remain connected. See "Using the Pen tool" on page 90.



A fill; the fill with a line drawn through it; and the two fills and three line segments created by segmentation

When you paint on top of shapes and lines, the portion underneath is replaced by whatever is on top. Paint of the same color merges together. Paint of different colors remains distinct. You can use these features to create masks, cutouts, and other negative images. For example, the cutout below was made by moving the ungrouped kite image onto the green shape, deselecting the kite, and then moving the filled portions of the kite away from the green shape.



To avoid inadvertently altering shapes and lines by overlapping them, you can group the shapes or use layers to separate them. See "Grouping objects" on page 146. For more information on layers, see "Using layers" in *Getting Started with Flash*.

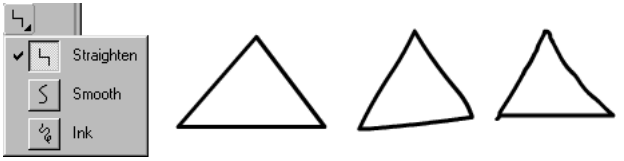
Drawing with the Pencil tool

To draw lines and shapes, you use the Pencil tool, in much the same way that you would use a real pencil to draw. To apply smoothing or straightening to the lines and shapes as you draw, you can select a drawing mode for the Pencil tool.

To draw with the Pencil tool:

1. Select the Pencil tool.
2. Select Window > Properties and select a stroke color, line weight, and style in the Property inspector. See "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.

3. Select a drawing mode under Options in the Tools panel:
   - Select Straighten to draw straight lines and convert approximations of triangles, ovals, circles, rectangles, and squares into these common geometric shapes.
   - Select Smooth to draw smooth curved lines.
   - Select Ink to draw freehand lines with no modification applied.



Lines drawn with Straighten, Smooth, and Ink mode, respectively

4. Drag on the Stage to draw with the Pencil tool. Shift-drag to constrain lines to vertical or horizontal directions.

Drawing straight lines, ovals, and rectangles

You can use the Line, Oval, and Rectangle tools to easily create these basic geometric shapes. The Oval and Rectangle tools create stroked and filled shapes. The Rectangle tool lets you create rectangles with square or rounded corners.

To draw a straight line, oval, or rectangle:

1. Select the Line, Oval, or Rectangle tool.
2. Select Window > Properties and select stroke and fill attributes in the Property inspector. See "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.

   Note: You cannot set fill attributes for the Line tool.

3. For the Rectangle tool, specify rounded corners by clicking the Round Rectangle modifier and entering a corner radius value. A value of zero creates square corners.
4. Drag on the Stage. If you are using the Rectangle tool, press the Up Arrow and Down Arrow keys while dragging to adjust the radius of rounded corners.

   For the Oval and Rectangle tools, Shift-drag to constrain the shapes to circles and squares.

   For the Line tool, Shift-drag to constrain lines to multiples of 45°.

Drawing polygons and stars

Using the PolyStar tool you can draw polygons or stars. You can choose the number of sides of the polygon or the number of points on the star, from 3 to 32. You can also choose the depth of the star points.

To draw a polygon or star:

1. Click and hold the mouse button on the Rectangle tool and drag to select the PolyStar tool from the pop-up menu.
2. Select Window > Properties to view the Property inspector.
3. Select stroke and fill attributes in the Property inspector. See "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.
4. Click the Options button in the Property inspector.
5. In the Tool Settings dialog box, do the following:
   - For Style, select Polygon or Star.
   - For Number of Sides, enter a number between 3 and 32.
   - For Star Point Size, enter a number between 0 and 1 to specify the depth of the star points. A number closer to 0 creates deeper points (like needles). If you are drawing a polygon, leave this setting unchanged. (It does not affect the polygon shape.)
6. Click OK to close the Tool Settings dialog box.
7. Drag on the Stage.

Using the Pen tool

To draw precise paths as straight lines or smooth, flowing curves, you can use the Pen tool. You can create straight or curved line segments and adjust the angle and length of straight segments and the slope of curved segments.

When you draw with the Pen tool, you click to create points on straight line segments, and click and drag to create points on curved line segments. You can adjust straight and curved line segments by adjusting points on the line. You can convert curves to straight lines and the reverse. You can also display points on lines that you create with other Flash drawing tools, such as the Pencil, Brush, Line, Oval, or Rectangle tools, to adjust those lines. See "Reshaping lines and shape outlines" on page 97.

Setting Pen tool preferences

You can specify preferences for the appearance of the Pen tool pointer, for previewing line segments as you draw, or for the appearance of selected anchor points. Selected line segments and anchor points are displayed using the outline color of the layer on which the lines and points appear.

To set Pen tool preferences:

1. Select the Pen tool, then select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Editing tab.
2. Under Pen Tool, set the following options:

   Show Pen Preview previews line segments as you draw. Flash displays a preview of the line segment as you move the pointer around the Stage, before you click to create the end point of the segment. If this option is not selected, Flash does not display a line segment until you create the end point of the segment.

   Show Solid Points displays selected anchor points as hollow and deselected anchor points as solid. If this option is not chosen, selected anchor points are solid, and deselected anchor points are hollow.

   Show Precise Cursors specifies that the Pen tool pointer appear as a cross-hair pointer, rather than the default Pen tool icon, for more precise placement of lines. Deselect the option to display the default Pen tool icon with the Pen tool.

   Note: Press the Caps Lock key while working to switch between the cross-hair pointer and the default Pen tool icon.

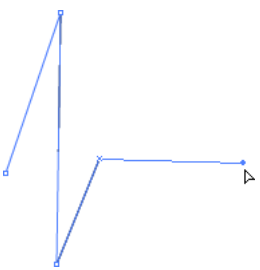
3. Click OK.

Drawing straight lines with the Pen tool

To draw straight line segments with the Pen tool, you create anchor points—points on the line that determine the length of individual line segments.

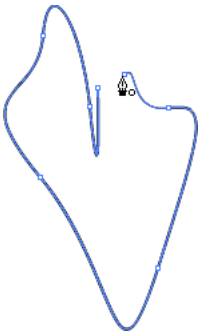
To draw straight lines with the Pen tool:

1. Select the Pen tool.
2. Select Window > Properties and select stroke and fill attributes in the Property inspector. See "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.
3. Position the pointer on the Stage where you want the straight line to begin, and click to define the first anchor point.
4. Click again where you want the first segment of the straight line to end. Shift-click to constrain lines to multiples of 45°.
5. Continue clicking to create additional straight segments.



6. To complete the path as an open or closed shape, do one of the following:

- To complete an open path, double-click the last point, click the Pen tool in the Tools panel, or Control-click (Windows) or Command-click (Macintosh) anywhere away from the path.
- To close a path, position the Pen tool over the first anchor point. A small circle appears next to the pen tip when it is positioned correctly. Click or drag to close the path.



- To complete the shape as is, select Edit > Deselect All or select a different tool in the Tools panel.

Drawing curved paths with the Pen tool

You create curves by dragging the Pen tool in the direction you want the curve to go to create the first anchor point, and then dragging the Pen tool in the opposite direction to create the second anchor point.

When you use the Pen tool to create a curved segment, the anchor points of the line segment display tangent handles. The slope and length of each tangent handle determine the slope and the height, or depth, of the curve. Moving the tangent handles reshapes the curves of the path. See "Adjusting segments" on page 94.

To draw a curved path:

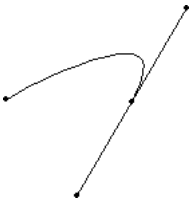
1. Select the Pen tool.
2. Position the Pen tool on the Stage where you want the curve to begin, and hold down the mouse button.

   The first anchor point appears, and the pen tip changes to an arrowhead.
3. Drag in the direction you want the curve segment to be drawn. Shift-drag to constrain the tool to multiples of 45°.

   As you drag, the tangent handles of the curve appear.
4. Release the mouse button.

   The length and slope of the tangent handles determine the shape of the curve segment. You can move the tangent handles later to adjust the curve.

5. Position the pointer where you want the curve segment to end, hold down the mouse button, and drag in the opposite direction to complete the segment. Shift-drag to constrain the segment to multiples of 45°.

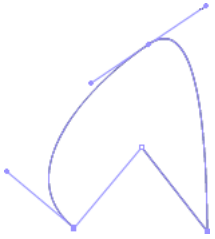


6. To draw the next segment of a curve, position the pointer where you want the next segment to end, and drag away from the curve.

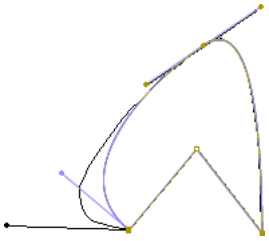
Adjusting anchor points on paths

When you draw a curve with the Pen tool, you create curve points—anchor points on a continuous, curved path. When you draw a straight line segment, or a straight line connected to a curved segment, you create corner points—anchor points on a straight path or at the juncture of a straight and a curved path.

By default, selected curve points appear as hollow circles, and selected corner points appear as hollow squares.



To convert segments in a line from straight segments to curve segments or the reverse, you convert corner points to curve points or the reverse.



You can also move, add, or delete anchor points on a path. You move anchor points using the Subselection tool to adjust the length or angle of straight segments or the slope of curved segments. You can nudge selected anchor points to make small adjustments.

Deleting unneeded anchor points on a curved path optimizes the curve and reduces the file size.

To move an anchor point:

- Drag the point with the Subselection tool.

To nudge an anchor point or points:

- Select the point or points with the Subselection tool and use the arrow keys to move the point or points.

To convert an anchor point, do one of the following:

- To convert a corner point to a curve point, use the Subselection tool to select the point, then Alt-drag (Windows) or Option-drag (Macintosh) the point to place the tangent handles.
- To convert a curve point to a corner point, click the point with the Pen tool.

To add an anchor point:

- Click a line segment with the Pen tool.

To delete an anchor point, do one of the following:

- To delete a corner point, click the point once with the Pen tool.
- To delete a curve point, click the point twice with the Pen tool. (Click once to convert the point to a corner point, and once more to delete the point.)
- Select the point with the Subselection tool and press Delete.

Adjusting segments

You can adjust straight segments to change the angle or length of the segment, or adjust curved segments to change the slope or direction of the curve.

When you move a tangent handle on a curve point, the curves on both sides of the point adjust. When you move a tangent handle on a corner point, only the curve on the same side of the point as the tangent handle adjusts.

To adjust a straight segment:

1. Select the Subselection tool, and select a straight segment.
2. Use the Subselection tool to drag an anchor point on the segment to a new position.

To adjust a curve segment:

- Select the Subselection tool and drag the segment.

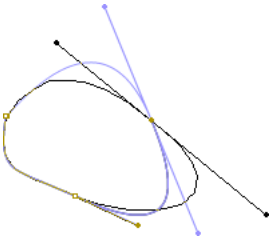
  Note: When you click the path, Flash shows the anchor points. Adjusting a segment with the Subselection tool may add points to the path.

To adjust points or tangent handles on a curve:

1. Select the Subselection tool, and select an anchor point on a curved segment.

   A tangent handle appears for the point you selected.

2. To adjust the shape of the curve on either side of the anchor point, drag the anchor point, or drag the tangent handle. Shift-drag to constrain the curve to multiples of 45°. Alt-drag (Windows) or Option-drag (Macintosh) to drag tangent handles individually.



Painting with the Brush tool

The Brush tool draws brushlike strokes, as if you were painting. It lets you create special effects, including calligraphic effects. You can select a brush size and shape using the Brush tool modifiers.

Brush size for new strokes remains constant even when you change the magnification level for the Stage, so the same brush size appears larger when the Stage magnification is lower. For example, suppose you set the Stage magnification to 100% and paint with the Brush tool using the smallest brush size. Then, you change the magnification to 50% and paint again using the smallest brush size. The new stroke that you paint appears 50% thicker than the earlier stroke. (Changing the magnification of the Stage does not change the size of existing brush strokes.)

You can use an imported bitmap as a fill when painting with the Brush tool. See "Breaking apart groups and objects" on page 155.

If you have a Wacom pressure-sensitive tablet connected to your computer, you can vary the width and angle of the brush stroke by using the Brush tool Pressure and Tilt modifiers, and varying pressure on the stylus.

The Pressure modifier varies the width of brush strokes when you vary the pressure on the stylus. The Tilt modifier varies the angle of brush strokes when you vary the angle of the stylus on the tablet. The Tilt modifier measures the angle between the top (eraser) end of the stylus and the top (north) edge of the tablet. For example, if you hold the pen vertically against the tablet, the Tilt is 90°. The Pressure and Tilt modifiers are both fully supported for the eraser function of the stylus.

A variable-width brush stroke drawn with a stylus

To paint with the Brush tool:

1. Select the Brush tool.
2. Select Window > Properties and select a fill color in the Property inspector. See "Using the Stroke Color and Fill Color controls in the Property inspector" on page 76.
3. Click the Brush Mode modifier and select a painting mode:

   Paint Normal paints over lines and fills on the same layer.

   Paint Fills paints fills and empty areas, leaving lines unaffected.

   Paint Behind paints in blank areas of the Stage on the same layer, leaving lines and fills unaffected.

   Paint Selection applies a new fill to the selection when you select a fill in the Fill modifier or the Fill box of the Property inspector. (This option is the same as simply selecting a filled area and applying a new fill.)

   Paint Inside paints the fill in which you start a brush stroke and never paints lines. This works much like a smart coloring book that never allows you to paint outside the lines. If you start painting in an empty area, the fill doesn't affect any existing filled areas.
4. Select a brush size and brush shape from the Brush tool modifiers.
5. If a Wacom pressure-sensitive tablet is attached to your computer, you can select the Pressure modifier, the Tilt modifier, or both, to modify brush strokes.
   - Select the Pressure modifier to vary the width of your brush strokes by varying the pressure on your stylus.
   - Select the Tilt modifier to vary the angle of your brush strokes by varying the angle of the stylus on the Wacom pressure-sensitive tablet.
6. Drag on the Stage. Shift-drag to constrain brush strokes to horizontal and vertical directions.

Reshaping lines and shape outlines

You can reshape lines and shape outlines created with the Pencil, Brush, Line, Oval, or Rectangle tools by dragging with the Selection tool, or by optimizing their curves.

You can also use the Subselection tool to display points on lines and shape outlines and modify the lines and outlines by adjusting the points. For information on adjusting anchor points, see "Using the Pen tool" on page 90.

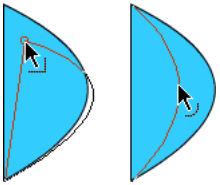
To display anchor points on a line or shape outline created with the Pencil, Brush, Line, Oval, or Rectangle tools:

1. Select the Subselection tool.
2. Click the line or shape outline.

Reshaping using the Selection tool

To reshape a line or shape outline, you can drag any point on a line using the Selection tool. The pointer changes to indicate what type of reshaping it can perform on the line or fill.

Flash adjusts the curve of the line segment to accommodate the new position of the moved point. If the repositioned point is an end point, you can lengthen or shorten the line. If the repositioned point is a corner, the line segments forming the corner remain straight as they become longer or shorter.



When a corner appears next to the pointer, you can change an end point. When a curve appears next to the pointer, you can adjust a curve.

Some brush stroke areas are easier to reshape if you view them as outlines.

If you are having trouble reshaping a complex line, you can smooth it to remove some of its details, making reshaping easier. Increasing the magnification can also make reshaping easier and more accurate; see "Optimizing curves" on page 99 or "Using the Stage" in *Getting Started with Flash*.

To reshape a line or shape outline using the Selection tool:

1. Select the Selection tool.
2. Do one of the following:
   - Drag from any point on the segment to reshape it.
   - Control-drag (Windows) or Option-drag (Macintosh) a line to create a new corner point.

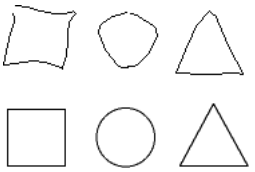
Straightening and smoothing lines

You can reshape lines and shape outlines by straightening or smoothing them.

Note: You can adjust the degree of automatic smoothing and straightening by specifying preferences for drawing settings. See "Specifying drawing settings" on page 103.

Straightening makes small straightening adjustments to lines and curves you have already drawn. It has no effect on already straight segments.

You can also use the straightening technique to make Flash recognize shapes. If you draw any oval, rectangular, or triangular shapes with the Recognize Shapes option turned off, you can use the Straightening option to make the shapes geometrically perfect. (For information on the Recognize Shapes option, see "Specifying drawing settings" on page 103.) Shapes that are touching, and thus connected to other elements, are not recognized.



Shape recognition turns the top shapes into the bottom shapes.

Smoothing softens curves and reduces bumps or other variations in a curve's overall direction. It also reduces the number of segments in a curve. Smoothing is relative, however, and has no effect on straight segments. It is particularly useful when you are having trouble reshaping a number of very short curved line segments. Selecting all the segments and smoothing them reduces the number of segments, producing a gentler curve that is easier to reshape.

Repeated application of smoothing or straightening makes each segment smoother or straighter, depending on how curved or straight each segment was originally.

To smooth the curve of each selected fill outline or curved line:

- Select the Selection tool and click the Smooth modifier in the Options section of the Tools panel, or select Modify > Smooth.

To make small straightening adjustments on each selected fill outline or curved line:

- Select the Selection tool and click the Straighten modifier in the Options section of the Tools panel, or select Modify > Shape > Straighten.

To use shape recognition:

- Select the Selection tool and click the Straighten modifier, or select Modify > Shape > Straighten.

Optimizing curves

Another way to smooth curves is to optimize them. This refines curved lines and fill outlines by reducing the number of curves used to define these elements. Optimizing curves also reduces the size of the Flash document (FLA file) and the exported Flash application (SWF file). As with the Smooth or Straighten modifiers or commands, you can apply optimization to the same elements multiple times.

To optimize curves:

1. Select the drawn elements to be optimized and select Modify > Optimize.
2. In the Optimize Curves dialog box, drag the Smoothing slider to specify the degree of smoothing.

   The exact results depend on the curves selected. Generally, optimizing produces fewer curves, with less resemblance to the original outline.
3. Set additional options:

   Use Multiple Passes repeats the smoothing process until no further optimization can be accomplished; this is the same as repeatedly selecting Optimize with the same elements selected.

   Show Totals Message displays an alert box that indicates the extent of the optimization when smoothing is complete.
4. Click OK.

Erasing

Erasing with the Eraser tool removes strokes and fills. You can quickly erase everything on the Stage, erase individual stroke segments or filled areas, or erase by dragging.

You can customize the Eraser tool to erase only strokes, only filled areas, or only a single filled area. The Eraser tool can be either round or square, and it can have one of five sizes.

To quickly delete everything on the Stage:

- Double-click the Eraser tool.

To remove stroke segments or filled areas:

1. Select the Eraser tool, and then click the Faucet modifier.
2. Click the stroke segment or filled area that you want to delete.

To erase by dragging:

1. Select the Eraser tool.
2. Click the Eraser Mode modifier and select an erasing mode:

   Erase Normal erases strokes and fills on the same layer.

   Erase Fills erases only fills; strokes are not affected.

   Erase Lines erases only strokes; fills are not affected.

Erase Selected Fills erases only the currently selected fills and does not affect strokes, selected or not. (Select the fills you want to erase before using the Eraser tool in this mode.)

Erase Inside erases only the fill on which you begin the eraser stroke. If you begin erasing from an empty point, nothing is erased. Strokes are unaffected by the eraser in this mode.

3. Click the Eraser Shape modifier and select an eraser shape and size. Make sure that the Faucet modifier is not selected.
4. Drag on the Stage.

Modifying shapes

You can modify shapes by converting lines to fills, expanding the shape of a filled object, or softening the edges of a filled shape by modifying the curves of the shape.

The Convert Lines to Fills feature changes lines to fills, which allows you to fill lines with gradients or to erase a portion of a line. The Expand Shape and Soften Edges features allow you to expand filled shapes and blur the edges of shapes.

The Expand Fill and Soften Fill Edges features work best on small shapes that do not contain many small details. Applying Soften Edges to shapes with extensive detail can increase the file size of a Flash document and the resulting SWF file.

To convert lines to fills:

1. Select a line or multiple lines.
2. Select Modify > Shape > Convert Lines to Fills.

   Selected lines are converted to filled shapes. Converting lines to fills can make file sizes larger, but it can also speed up drawing for some animations.

To expand the shape of a filled object:

1. Select a filled shape. This command works best on a single filled color shape with no stroke.
2. Select Modify > Shape > Expand Fill.
3. In the Expand Path dialog box, enter a value in pixels for Distance and select Expand or Inset for Direction. Expand enlarges the shape, and Inset reduces it.

To soften the edges of an object:

1. Select a filled shape.

   Note: This feature works best on a single filled shape that has no stroke.

2. Select Modify > Shape > Soften Fill Edges.
3. Set the following options:

   Distance is the width, in pixels, of the soft edge.

   Number of Steps controls how many curves are used for the soft edge effect. The more steps you use, the smoother the effect. Increasing steps also creates larger files and slows drawing.

   Expand or Inset controls whether the shape is enlarged or reduced to soften the edges.

Snapping

To automatically align elements with one another, you can use snapping. Flash provides three ways for you to align objects on the Stage:

- Object snapping lets you snap objects directly to other objects along their edges.
- Pixel snapping lets you snap objects directly to individual pixels or lines of pixels on the Stage.
- Snap alignment lets you snap objects to a specified *snap tolerance*, a preset boundary between objects and other objects or between objects and the edge of the Stage.

Note: You can also snap to the grid or to guides. For more information, see "About the main toolbar and edit bar" in *Getting Started with Flash*.

Object snapping

Object snapping can be turned on using the Snap modifier for the Selection tool, or the Snap to Objects command in the View menu.

If the Snap modifier for the Selection tool is on, a small black ring appears under the pointer when you drag an element. The small ring changes to a larger ring when the object is within snapping distance of another object.

To turn object snapping on or off:

- Select View > Snapping > Snap to Objects. A check mark is displayed next to the command when it is on.

  When you move or reshape an object, the position of the Selection tool on the object provides the reference point for the snap ring. For example, if you move a filled shape by dragging near its center, the center point snaps to other objects. This is particularly useful for snapping shapes to motion paths for animating.

  Note: For better control of object placement when snapping, begin dragging from a corner or center point.

To adjust object snapping tolerances:

1. Select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Editing tab.
2. Under Drawing Settings, adjust the Connect Lines setting. See "Specifying drawing settings" on page 103.

Pixel snapping

You can turn on pixel snapping using the Snap to Pixels command in the View menu. If Snap to Pixels is on, a pixel grid appears when the view magnification is set to 400% or higher. The pixel grid represents the individual pixels that will appear in your Flash application. When you create or move an object, it is constrained to the pixel grid.

If you create a shape whose edges fall between pixel boundaries—for example, if you use a stroke with a fractional width, such as 3.5 pixels—keep in mind that Snap to Pixels snaps to pixel boundaries, and not to the edge of the shape.

To turn pixel snapping on or off:

- Select View > Snapping > Snap to Pixels.

  If the magnification is set to 400% or higher, a pixel grid is displayed. A check mark is displayed next to the command when it is on.

To turn pixel snapping on or off temporarily:

- Press the C key. When you release the C key, pixel snapping returns to the state you selected with View > Snapping > Snap to Pixels.

To temporarily hide the pixel grid:

- Press the X key. When you release the X key, the pixel grid reappears.

Snap alignment

You can turn on Snap Alignment using the Snap Align command in the View menu. You can select settings for Snap Alignment using the Edit Snap Align command in the View menu.

When you select Snap Alignment settings, you can set the snap tolerance between horizontal or vertical edges of objects, and between objects' edges and the Stage border. You can also turn on snap alignment between the horizontal and the vertical centers of objects. All Snap Alignment settings are measured in pixels.

When Snap Alignment is turned on, dotted lines appear on the Stage when you drag an object to the specified snap tolerance. For example, if you set Horizontal snap tolerance to 18 pixels (the default setting), a dotted line appears along the edge of the object you are dragging when the object is exactly 18 pixels from another object. If you turn on Horizontal Center Alignment, a dotted line appears along the horizontal center vertices of two objects when you precisely align the vertices.

To select settings for Snap Alignment:

1. Select View > Snapping > Edit Snap Align.
2. In the Snap Align dialog box, do any of the following:
   - To set the snap tolerance between objects and the Stage border, enter a value for Movie Border.
   - To set the snap tolerance between horizontal or vertical edges of objects, enter a value for Horizontal, Vertical, or both.
   - To turn on Horizontal or Vertical Center Alignment, select Horizontal or Vertical Center Alignment or both.

To turn on Snap Alignment:

- Select Snapping > Snap Align.

Specifying drawing settings

You can set drawing settings to specify snapping, smoothing, and straightening behaviors when you use Flash drawing tools. You can change the tolerance setting for each option, and turn each option off or on. Tolerance settings are relative, depending on the resolution of your computer screen and the current magnification of the scene. By default, each option is turned on and set to Normal tolerance.

To set drawing settings:

1. Select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Editing tab.
2. Under Drawing Settings, select from the following options:

   Connect Lines determines how close the end of a line being drawn must be to an existing line segment before the end point snaps to the nearest point on the other line. The available options are Must Be Close, Normal, and Can Be Distant. This setting also controls horizontal and vertical line recognition—that is, how nearly horizontal or vertical a line must be drawn before Flash makes it exactly horizontal or vertical. When Snap to Objects is turned on, this setting controls how close objects must be to snap to one another.

   Smooth Curves specifies the amount of smoothing applied to curved lines drawn with the Pencil tool when the drawing mode is set to Straighten or Smooth. (Smoother curves are easier to reshape, whereas rougher curves match the original line strokes more closely.) The selections are Off, Rough, Normal, and Smooth.

   Note: You can further smooth existing curved segments using Modify › Shape › Smooth and Modify › Shape › Optimize.

   Recognize Lines defines how nearly straight a line segment drawn with the Pencil tool must be before Flash recognizes it as a straight line and makes it perfectly straight. The selections are Off, Strict, Normal, and Tolerant. If Recognize Lines is off while you draw, you can straighten lines later by selecting one or more line segments and selecting Modify > Shape > Straighten.

   Recognize Shapes controls how precisely you must draw circles, ovals, squares, rectangles, and 90° and 180° arcs for them to be recognized as geometric shapes and redrawn accurately. The options are Off, Strict, Normal, and Tolerant. If Recognize Shapes is off while you draw, you can straighten lines later by selecting one or more shapes (for example, connected line segments) and selecting Modify > Shape > Straighten.

   Click Accuracy specifies how close to an item the pointer must be before Flash recognizes the item. The options are Strict, Normal, and Tolerant.

CHAPTER 6
Working with Text

You can include text in your Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 applications in a variety of ways. You can create text blocks containing *static* text, text whose contents and appearance you determine when you author the document. You can also create *dynamic* or *input* text fields. Dynamic text fields display dynamically updating text, such as sports scores or stock quotes. Input text fields allow users to enter text for forms, surveys, or other purposes.

Just like movie clip instances, text field instances are ActionScript objects that have properties and methods. By giving a text field an instance name, you can manipulate it with ActionScript. However, unlike with movie clips, you cannot write ActionScript code inside a text instance, because text instances don't have Timelines.

You can orient text horizontally, with left-to-right flow, or vertically (static text only), with left-to-right or right-to-left flow. You can select the following attributes for text: font, point size, style, color, tracking, kerning, baseline shift, alignment, margins, indents, and line spacing. See "Setting text attributes" on page 111.

The Check Spelling feature lets you to check spelling in text fields, as well as in scene and layer names, frame labels, ActionScript strings, and other places where text occurs in your document. See "Checking spelling" on page 117.

You can transform text as you would an object—rotating, scaling, skewing, and flipping it—and still edit its characters. See "About transforming text" on page 119. When you're working with horizontal text, you can link text blocks to URLs and make it selectable. See "Linking text to a URL (horizontal text only)" on page 120.

Timeline effects let you apply prebuilt animation effects to text, such as bouncing, fading in or out, and exploding. See "Using Timeline effects with text" on page 119.

When you work with Flash FLA files, Flash substitutes fonts in the FLA file with other fonts installed on your system if the specified fonts are not on your system. You can select options to control which fonts are used in substitution. Substitute fonts are used for display on your system only. The font selection in the FLA file remains unchanged. See "Substituting missing fonts" on page 122.

Flash also lets you create a symbol from a font so that you can export the font as part of a shared library and use it in other Flash documents. See "Creating font symbols" on page 116.

You can break text apart and reshape its characters. For additional text-handling capabilities, you can manipulate text in FreeHand and import the FreeHand file into Flash, or export the file from FreeHand as a SWF file. See "Breaking text apart" on page 120.

Flash documents can use Type 1 PostScript fonts, TrueType, and bitmap fonts (Macintosh only). You can check spelling by copying text to the Clipboard using the Movie Explorer and pasting the text into an external text editor. See "Using the Movie Explorer" on page 28.

You can preserve rich text formatting in text fields, using HTML tags and attributes. See "Preserving rich text formatting" on page 121.

When you use HTML text for the content of a dynamic or input text field, you can flow the text around an image, including a SWF or JPG file or a movie clip. See "Using HTML-formatted text" in *Using ActionScript in Flash*.

You can use ActionScript to format input and dynamic text, and to create scrolling text fields. ActionScript has events for dynamic and input text fields that you can capture and use to trigger scripts. For information on using ActionScript to control text, see Chapter 9, "Working with Text," in *Using ActionScript in Flash*.

For an interactive introduction to creating text in Flash, select Help > How Do > Basic Flash > Add Static, Input, and Dynamic Text.

This chapter contains the following sections:

About Unicode text encoding in Flash applications

Macromedia Flash Player 7 supports Unicode text encoding for SWF files in Macromedia Flash Player 7 format. This support greatly enhances your ability to use multilingual text in SWF files that you create with Flash, including multiple languages within a single text field. Any user with Macromedia Flash Player 7 can view multilanguage text in a Macromedia Flash Player 7 application, regardless of the language used by the operating system running the player.

For information on Unicode support in Macromedia Flash, see Chapter 13, "Creating Multilanguage Text," on page 235.

About font outlines and device fonts

When you publish or export a Flash application containing static text, Flash creates outlines of the text and uses the outlines to display the text in Flash Player.

When you publish or export a Flash application containing dynamic or input text fields, Flash stores the names of the fonts used in creating the text. Flash Player uses the font names to locate identical or similar fonts on the user's system when the Flash application is displayed. You can also export font outlines with dynamic or input text by clicking the Character option in the Property inspector and selecting options. See "Setting dynamic and input text options" on page 115.

Not all fonts displayed in Flash can be exported as outlines with a Flash application. To verify that a font can be exported, you can use the View > Preview Mode > Antialias Text command to preview the text; jagged type indicates that Flash does not recognize that font's outline and will not export the text.

About using device fonts

For static horizontal text only, you can use special fonts in Flash called *device fonts* as an alternative to exporting font outline information. Device fonts are not embedded in the Flash SWF file. Instead, Flash Player uses whatever font on the local computer most closely resembles the device font. Because device font information is not embedded, using device fonts results in a somewhat smaller SWF file. In addition, device fonts can be sharper and more legible than exported font outlines at small point sizes (below 10 points). However, because device fonts are not embedded, text may look different than expected in user systems that do not have an installed font corresponding to the device font.

Flash includes three device fonts, named _sans (similar to Helvetica or Arial), _serif (similar to Times Roman), and _typewriter (similar to Courier). To specify a font as a device font, you select one of the Flash device fonts in the Property inspector. During SWF file playback, Flash selects the first device font that is located on the user's system. See "Making text selectable by users" on page 114.

About masking device fonts

You can use a movie clip to mask text that is set in a device font and converted into a movie clip. For a movie clip mask on a device font to function, the user must have Flash Player 6 (6.0.40.0) or later.

When you use a movie clip to mask text set in a device font, the rectangular bounding box of the mask is used as the masking shape. That is, if you create a nonrectangular movie clip mask for device font text in the Flash authoring environment, the mask that appears in the SWF file takes the shape of the rectangular bounding box of the mask, not the shape of the mask itself.

You can mask device fonts only by using a movie clip as a mask. You cannot mask device fonts by using a mask layer on the Stage.

For more information on using a movie clip as a mask, see "Using movie clips as masks" in *Using ActionScript in Flash*.

Creating text

You can create three types of text fields: static, dynamic, and input. All text fields support Unicode.

- Static text fields display text that doesn't change characters dynamically.
- Dynamic text fields display dynamically updating text, such as sports scores, stock quotes, or weather reports.
- Input text fields allow users to enter text in forms or surveys.

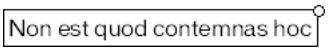
In Flash, you can create horizontal text (with a left-to-right flow) or static vertical text (with either a right-to-left or left-to-right flow). By default, text is created with horizontal orientation. You can select preferences to make vertical text the default orientation and to set other options for vertical text.

You can also create scrolling text fields. See "Creating scrolling text" on page 128.

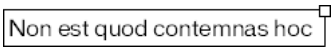
To create text, you place text blocks on the Stage using the Text tool. When creating static text, you can place text on a single line that expands as you type, or in a fixed-width block (for horizontal text) or fixed-height block (for vertical text) that expands and wraps words automatically. When creating dynamic or input text, you can place text on a single line, or create a text block with a fixed width and height.

Flash displays a handle on the corner of a text block to identify the type of text block:

- For static horizontal text that extends, a round handle appears at the upper right corner of the text block.



- For static horizontal text with a defined width, a square handle appears at the upper right corner of the text block.



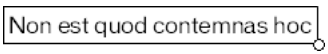
- For static vertical text that has right-to-left orientation and extends, a round handle appears at the lower left corner of the text block.

- For static vertical text with right-to-left orientation and a fixed height, a square handle appears at the lower left corner of the text block.

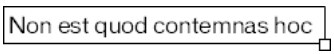
- For static vertical text that has left-to-right orientation and extends, a round handle appears at the lower right corner of the text block.

- For static vertical text with left-to-right orientation and a fixed height, a square handle appears at the lower right corner of the text block.

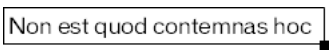
- For dynamic or input text blocks that extend, a round handle appears at the lower right corner of the text block.



- For dynamic or input text with a defined height and width, a square handle appears at the lower right corner of the text block.



- For dynamic scrollable text blocks, the round or square handle becomes solid black instead of hollow. See "Creating scrolling text" on page 128.



You can Shift-double-click the handle of dynamic and input text fields to create text blocks that don't expand when you enter text on the Stage. This allows you to create a text block of a fixed size and fill it with more text than it can display to create scrolling text. See "Creating scrolling text" on page 128.

After you use the Text tool to create a text field, you use the Property inspector to indicate which type of text field you want and set values to control how the text field and its contents appear in the SWF file.

To set preferences for vertical text:

1. Select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Editing tab in the Preferences dialog box.
2. Under Vertical Text, select Default Text Orientation to automatically give new text blocks vertical orientation.
3. Select Right to Left Text Flow to make vertical text automatically flow right to left.
4. Select No Kerning to prevent kerning from being applied to vertical text. (Kerning remains enabled for horizontal text.) For more information on kerning, see"Setting character spacing, kerning, and character position" on page 113.

To create text:

1. Select the Text tool.
2. Select Window > Properties.
3. In the Property inspector, select a text type from the pop-up menu to specify the type of text field:

   Dynamic Text creates a field that displays dynamically updating text.

   Input Text creates a field in which users can enter text.

   Static Text creates a field that cannot update dynamically.

4. For static text only: In the Property inspector, click the Text Direction button (in the top row, to the right of the Italic button) and select an option to specify the orientation of the text:

   Horizontal makes text flow horizontally, left to right (the default setting).

   Vertical Left-to-Right makes text flow vertically, left to right.

   Vertical Right-to-Left makes text flow vertically, right to left.

   Note: Layout options for vertical text are disabled if the text is dynamic or input. Only static text can be vertical.

5. Do one of the following:
   - To create a text block that displays text in a single line, click where you want the text to start.
   - To create a text block with a fixed width (for horizontal text) or fixed height (for vertical text), position the pointer where you want the text to start and drag to the desired width or height.

   Note: If you create a text block that extends past the edge of the Stage as you type, the text isn't lost. To make the handle accessible again, add line breaks, move the text block, or select View > Work Area.

6. Select text attributes in the Property inspector as described in "Setting text attributes" on page 111.

To change the dimensions of a text block:

- Drag its resize handle.

To switch a text block between fixed-width or fixed-height and extending:

- Double-click the resize handle.

Creating scrolling text

There are several ways to create scrolling text in Flash. You can easily make dynamic text fields scrollable using menu commands or the text block handle.

You can also add a ScrollBar component to a text field to make it scroll. For more information, see "UIScrollBar component" in *Using Components*.

If you want to use ActionScript, you can use the `scroll` and `maxscroll` properties of the TextField object to control vertical scrolling and the `hscroll` and `maxhscroll` properties to control horizontal scrolling in a text block. See "Creating scrolling text" in *Using ActionScript in Flash*.

To make a dynamic text block scrollable, do one of the following:

- Shift-double-click the handle on the dynamic text block.
- Select the dynamic text block with the Selection tool and select Text > Scrollable.
- Select the dynamic text block with the Selection tool. Right-click (Windows) or Control-click (Macintosh) the dynamic text block and select Text > Scrollable.

Setting text attributes

You can set the font and paragraph attributes of text. Font attributes include font family, point size, style, color, character spacing, autokerning, and character position. Paragraph attributes include alignment, margins, indents, and line spacing.

You can optimize text to make text more readable at small sizes. See "About aliasing text" on page 111.

For static text, font outlines are exported in a published SWF file. You can choose to use device fonts, rather than exporting font outlines (horizontal text only). See "About font outlines and device fonts" on page 107.

For dynamic or input text, Flash stores the names of the fonts used in creating the text. Flash Player uses the names to locate identical or similar fonts on the user's system when the Flash application is playing. You can also choose to embed font outlines in dynamic or input text fields. Embedding font outlines can increase file size, but it ensures that users have the correct font information. See "Setting dynamic and input text options" on page 115.

When text is selected, you use the Property inspector to change font and paragraph attributes, and to direct Flash to use device fonts rather than embedding font outline information.

When creating new text, Flash uses the current text attributes. To change the font or paragraph attributes of existing text, you must first select the text.

About aliasing text

The Alias Text button in the Property inspector lets you render text so that it appears more readable at small sizes. This option is supported for static, dynamic, and input text if the end user has Flash Player 7 or later. It is supported only for static text if the user has an earlier version of Flash Player. See "Choosing a font, point size, style, and color" on page 112.

The Alias Text option makes small text more readable by aligning text outlines along pixel boundaries. This makes the text appear aliased, even when anti-aliasing is enabled. For information on anti-aliasing text, see "Speeding up document display" on page 40.

When Alias Text is enabled, all text in the current selection is affected. The feature operates with text of all point sizes in the same way.

When using small text in a Flash document, keep in mind the following guidelines:

- Very small text (below 8 points) may not be displayed clearly, even with Alias Text selected.
- Sans serif text, such as Helvetica or Arial, appears clearer at small sizes than serif text.
- Some type styles, such as bold and italic, can reduce readability of text at small sizes.
- In some cases, text in Flash appears somewhat smaller than text of the same point size in other applications.

Choosing a font, point size, style, and color

You can set the font, point size, style, and color for selected text using the Property inspector. When setting the text color, you can use only solid colors, not gradients. To apply a gradient to text, you must convert the text to its component lines and fills. See "Breaking text apart" on page 120.

To select a font, point size, style, and color with the Property inspector:

1. Select the Text tool.
2. To apply settings to existing text, use the Text tool to select a text block or text blocks on the Stage.
3. If the Property inspector is not visible, select Window > Properties.
4. In the Property inspector, click the triangle next to the Font text box and select a font from the list, or enter a font name.

   Note: The fonts _sans, _serif, and _typewriter are device fonts. Font outline information for these fonts is not embedded in the Flash SWF file. Device fonts can be used only with horizontal text. See "About font outlines and device fonts" on page 107.

5. Click the triangle next to the Point Size value and drag the slider to select a value, or enter a font size value.

   Text size is set in points, regardless of the current ruler units.
6. To apply bold or italic style, click the Bold button or the Italic button.
7. Click the Alias Text button (directly below the Bold button) to optimize text.
8. To select a fill color for text, click the color box and do one of the following:
   - Select a color from the color pop-up window.
   - Type a color's hexadecimal value in the text box in the color pop-up window.
   - Click the Color Picker button in the upper right corner of the pop-up window and select a color from the system Color Picker.

   For more information on selecting colors, see Chapter 4, "Working with Color," on page 75.

Setting character spacing, kerning, and character position

Character spacing inserts a uniform amount of space between characters. You use character spacing to adjust the spacing of selected characters or entire blocks of text.

Kerning controls the spacing between pairs of characters. Many fonts have built-in kerning information. For example, the spacing between an *A* and a *V* is often less than the spacing between an *A* and a *D*. To use a font's built-in kerning information to space characters, you use the Kern option.

For horizontal text, tracking and kerning set the horizontal distance between characters. For vertical text, tracking and kerning set the vertical distance between characters.

For vertical text, you can set kerning to be off by default in Flash Preferences. When kerning is turned off for vertical text in Preferences, you can leave the option selected in the Property inspector, and kerning will be applied to horizontal text only. To set preferences for vertical text, see "Creating text" on page 108.

Using the Property inspector, you can also apply superscript or subscript styles to your text.

To set character spacing, kerning, and character position:

1. Select the Text tool.
2. To apply settings to existing text, use the Text tool to select a text block or text blocks on the Stage.
3. If the Property inspector is not already displayed, select Window > Properties.
4. In the Property inspector, set the following options:
   - To specify character spacing, click the triangle next to the Character Spacing value and drag the slider to select a value, or enter a value in the text box.
   - To use a font's built-in kerning information, select Kern.
   - To specify character position, click the triangle next to the Character Position option and select a position from the menu: Normal places text on the baseline, Superscript places text above the baseline (horizontal text) or to the right of the baseline (vertical text), and Subscript places text below the baseline (horizontal text) or to the left of the baseline (vertical text).

Setting alignment, margins, indents, and line spacing

Alignment determines the position of each line of text in a paragraph relative to edges of the text block. Horizontal text is aligned relative to the left and right edges of the text block, and vertical text is aligned relative to the top and bottom edges of the text block. Text can be aligned to one edge of the text block, centered in the text block, or aligned to both edges of the text block (full justification).

Margins determine the amount of space between the border of a text block and a paragraph of text. Indents determine the distance between the margin of a paragraph and the beginning of the first line. For horizontal text, indents move the first line to the right the specified distance. For vertical text, indents move the first line down the specified distance.

Line spacing determines the distance between adjacent lines in a paragraph. For vertical text, line spacing adjusts the space between vertical columns.

To set alignment, margins, indents, and line spacing for horizontal text:

1. Select the Text tool.
2. To apply settings to existing text, use the Text tool to select a text block or text blocks on the Stage.
3. Select Window > Properties.
4. In the Property inspector, click Format Options and set the following options:
   - To set alignment, click the Left, Center, Right, or Full Justification button.
   - To set left or right margins, click the triangle next to the Left Margin or Right Margin value and drag the slider to select a value, or enter a value in the numeric field.
   - To specify indents, click the triangle next to the Indent value and drag the slider to select a value, or enter a value in the numeric field. (Either the right or left line is indented, depending on whether the text flows right to left or left to right.)
   - To specify line spacing, click Format options. Click the triangle next to the Line Spacing value and drag the slider to select a value, or enter a value in the numeric field.

To set alignment, margins, indents, and line spacing for vertical text:

1. Select the Text tool.
2. To apply settings to existing text, select a text block or text blocks on the Stage.
3. Select Window > Properties.
4. In the Property inspector, click Format Options and set the following options:
   - To set alignment, click the Top, Center, Bottom, or Full Justification button.
   - To set top or bottom margins, use the Left or Right margin control. Click the triangle next to the Left Margin value to set the top margin or the Right Margin value to set the bottom margin and drag the slider to select a value, or enter a value in the numeric field.
   - To specify indents, click the triangle next to the Indent value and drag the slider to select a value, or enter a value in the numeric field.
   - To specify line spacing, click the triangle next to the Line Spacing value and drag the slider to select a value, or enter a value in the numeric field.

Making text selectable by users

When working with static horizontal text, you can specify that fonts be selectable by users viewing your Flash application. After selecting text, the user can copy, cut, and then paste the text into a new document.

To make horizontal text selectable by a user:

1. Select the horizontal text that you want to make selectable by a user.
2. Select Window > Properties.
3. In the Property inspector, select Static Text or Dynamic Text (Input Text is selectable by default).
4. Click the Selectable button.

Using device fonts (static horizontal text only)

When you create static text, you can specify that Flash Player use device fonts to display certain text blocks. Using device fonts can decrease the file size of your document, because the document does not contain font outlines for the text. Using device fonts can also increase legibility at text sizes below 10 points.

You can use movie clips to mask text set in device fonts. See "About masking device fonts" on page 107.

To specify that text be displayed using a device font:

1. Select text blocks on the Stage containing text that you want to display using a device font.
2. Select Window > Properties.
3. In the Property inspector, select Static Text from the pop-up menu.
4. Select Use Device Fonts.

Setting dynamic and input text options

The Property inspector lets you specify options that control how dynamic and input text appears in the Flash application.

To set options for dynamic and input text:

1. Click in an existing dynamic text field.

   To create a new dynamic text field, see "Creating text" on page 108.
2. In the Property inspector, make sure Dynamic or Input is displayed in the pop-up menu. Do any of the following:
   - For Instance Name, enter the instance name for the text field.
   - Lock height, width, and location of text.
   - Select font type and style.
   - Select Multiline to display the text in multiple lines, Single Line to display the text as one line, or Multiline No Wrap to display text in multiple lines that break only if the last character is a breaking character, such as Enter (Windows) or Return (Macintosh).
   - Click the Selectable button to enable users to select dynamic text. Deselect this option to prevent users from selecting the dynamic text.

- Click the Render Text as HTML button to preserve rich text formatting, such as fonts and hyperlinks, with the appropriate HTML tags. See "Preserving rich text formatting" on page 121.
- Click the Show Border button to display a black border and white background for the text field.
- For Variable, enter the variable name for the text field.
- Select Character for embedded font outlines options. In the Character Options dialog box, click No characters not to embed font outlines or Specify Range to embed fonts outlines. When Specify Range is selected, you can select one or more options from the scrolling list, type only the characters to embed in the document, or click Auto Fill to copy each unique character from the selected text to the text box. Then click OK.

Creating font symbols

To use a font as a shared library item, you can create a font symbol in the Library panel. You then assign the symbol an identifier string and a URL where the document containing the font symbol will be posted. In this way, you can link to the font and use it in a Flash application.

Note: When using font symbols for dynamic or input text, you must also embed the font outline information. See "Setting dynamic and input text options" on page 115.

For information on linking to a shared font symbol from other documents, see "Using shared library assets" on page 69.

To create a font symbol:

1. Open the library to which you want to add a font symbol.
2. Select New Font from the options menu in the upper right corner of the Library panel.
3. In the Font Symbol Properties dialog box, enter a name for the font symbol in the Name text box.
4. Select a font from the Font menu or enter the name of a font in the Font text box.
5. If you want to apply a style to the font, select Bold or Italic.
6. Click OK.

To assign an identifier string to a font symbol:

1. Select the font symbol in the Library panel.
2. Do one of the following:
   - Select Linkage from the options menu in the upper right corner of the Library panel.
   - Right-click (Windows) or Control-click (Macintosh) the font symbol name in the Library panel, and select Linkage from the context menu.
3. Under Linkage in the Linkage Properties dialog box, select Export for Runtime Sharing.
4. In the Identifier text box, enter a string to identify the font symbol.

5. In the URL text box, enter the URL where the SWF file that contains the font symbol will be posted.
6. Click OK.

To use the font in a Flash application, copy the font symbol into the destination FLA file. For more information, see "Copying library assets between documents" on page 68.

Editing text

You can use most common word-processing techniques to edit text in Flash. You use the Cut, Copy, and Paste commands to move text in a Flash file as well as between Flash and other applications.

When editing text or changing text attributes, you must first select the characters you want to change.

To select characters in a text block:

1. Select the Text tool.
2. Do one of the following:
   - Drag to select characters.
   - Double-click to select a word.
   - Click to specify the beginning of the selection and Shift-click to specify the end of the selection.
   - Press Control+A (Windows) or Command+A (Macintosh) to select all the text in the block.

To select text blocks:

- Select the Selection tool and click a text block. Shift-click to select multiple text blocks.

Checking spelling

The Check Spelling feature enables you to check spelling in text throughout your Flash document.

You can use Spelling Setup to select a variety of options for spell checking:

- Select document options to specify which elements in a Flash document are to be checked, including text fields, scene and layer names, frame labels and comments, and others.
- Select one or more built-in dictionaries to use when checking spelling.
- Create a personal dictionary with words and phrases you add yourself.
- Select checking options to specify ways to handle specific word and character types, such as nonalphabetic words or Internet addresses, when spell checking.

When the Check Spelling feature identifies a word not found in the specified dictionary or dictionaries, you can select how to handle the word:

- Change the identified word or all occurrences of the word.
- Select a suggested word to use in changing the identified word.

- Ignore the identified word or all occurrences of the word.
- Add the identified word to your personal dictionary.
- Delete the identified word.

Using Spelling Setup

You use the Spelling Setup dialog box to specify options for the Check Spelling feature. Before you check spelling the first time, you must specify spelling options in the Spelling Setup dialog box to initialize the Check Spelling feature. After you initialize Check Spelling, you can use the Spelling Setup dialog box to change options for spell checking.

To use Spelling Setup:

1. Open the Spelling Setup dialog box. Do one of the following:
   - Select Text > Spelling Setup. (Use this option if you have not initialized the Check Spelling feature before.)
   - In the Check Spelling dialog box (Text > Check Spelling), click the Setup button.
2. In the Spelling Setup dialog box, select any of the items in the Document Options list to specify document-level spell checking options. You can select options to check spelling in specified text sources in a document, to select the text item during spell checking, and to enable live edit on the text item during spell checking.
3. In the Dictionaries scroll list, select one or more dictionaries from the Macromedia dictionaries installed with your product. You must select at least one dictionary to enable spell checking.
4. Under Personal Dictionary, enter a path or click the folder icon and browse to a document that you want to use as a personal dictionary.
5. To add words and phrases to your personal dictionary, click Edit Personal Dictionary. In the Personal Dictionary dialog box, enter each new item on a separate line in the text field. Click OK to save the items and close the dialog box.
6. Select any of the items under Checking Options to specify word-level spell checking options. You can select options to ignore specific word or character types, to find duplicate words, to split contracted or hyphenated words, or to suggest phonetic or typographical matches.
7. Click OK to save the settings and exit Spelling Setup.

Using the Check Spelling feature

To check spelling of text in a document, you use the Check Spelling feature, which checks spelling based on options you select in Spelling Setup. When the spell checker identifies a word not found in the dictionary or dictionaries, you can choose to change, ignore, or delete the word, or add it to the personal dictionary.

To use the Check Spelling feature:

1. Select Text > Check Spelling to view the Check Spelling dialog box.

   The text box in the upper left corner identifies words not found in the selected dictionary or dictionaries, and also identifies the type of element where the text is located (such as text field, frame label, or other).

2. Do one of the following:
   - Click the Add to Personal button to add the word to your personal dictionary.
   - Click Ignore to leave the word unchanged. Click Ignore All to leave all occurrences of the word in the document unchanged.
   - Enter a word in the Change To text box or select a word from the Suggestions scroll list. Then click Change to change the word or click Change All to change all occurrences of the word in the document.
   - Click Delete to delete the word from the document.
3. To change Spelling Setup options, click Setup.
4. To end spell checking, do one of the following:
   - Click Close to end spell checking before Flash reaches the end of the document.
   - Continue spell checking until you see a notification that Flash has reached the end of the document, then click No to end spell checking. Click Yes to resume spell checking at the beginning of the document.

About transforming text

You can transform text blocks in the same ways that you transform other objects. You can scale, rotate, skew, and flip text blocks to create interesting effects. When you scale a text block as an object, increases or decreases in point size are not reflected in the Property inspector.

The text in a transformed text block can still be edited, although severe transformations may make it difficult to read.

For more information about transforming text blocks, see Chapter 8, “Working with Graphic Objects,” on page 143.

Using Timeline effects with text

You can use Timeline effects to easily add animation to text. Timeline effects are prebuilt animation effects that let you add motion to text with a minimum of effort. For example, you can use Timeline effects to make text bounce, fade in or out, or explode. For more information on using each effect, see “Using Timeline effects” on page 158.

Breaking text apart

You can break apart text to place each character in a separate text block. After you break text apart, you can quickly distribute the text blocks to separate layers and animate each block separately. For information on distributing objects to layers, see "Distributing objects to layers for tweened animation" on page 164. For general information on animation, see Chapter 9, "Creating Motion," on page 157.

Note: You cannot break apart text in scrollable text fields.

You can also convert text to its component lines and fills to reshape, erase, and otherwise manipulate it. As with any other shape, you can individually group these converted characters, or change them to symbols and animate them. After you convert text to lines and fills, you can no longer edit the text.

To break apart text:

1. Select the Selection tool and click a text block.
2. Select Modify > Break Apart. Each character in the selected text is placed in a separate text block. The text remains in the same position on the Stage.
3. Select Modify > Break Apart again to convert the characters to shapes on the Stage.

   Note: The Break Apart command applies only to outline fonts such as TrueType fonts. Bitmap fonts disappear from the screen when you break them apart. PostScript fonts can be broken apart only on Macintosh systems.

Linking text to a URL (horizontal text only)

You can link horizontal text to a URL so that users can jump to other files by clicking the text.

To link horizontal text to a URL:

1. Select some text or a text block. Do one of the following:
   - Use the Text tool to select text in a text block.
   - Use the Selection tool to select a text block on the Stage. This links all the text in the block to a URL.
2. If the Property inspector is not already displayed, select Window > Properties.
3. For Link, enter the URL to which you want to link the text block.

   Note: To create a link to an e-mail address, use the mailto: URL. For example, for the Macromedia Flash Wish URL, enter **mailto:wish-flash@macromedia.com**.

Preserving rich text formatting

Flash lets you preserve rich text formatting in input and dynamic text fields. If you select the Render Text as HTML formatting option in the Property inspector or set the `html` property of the TextField object to `true`, Flash preserves basic text formatting (such as font, style, color, and size) and hyperlinks in the text field by automatically applying the corresponding HTML tags when you export the SWF file. You apply HTML tags to text fields as the value of the `htmlText` property of the TextField object. You must give the text field an instance name to use the `htmlText` property.

If you will publish your Flash document as Flash Player 5 or earlier, you can use the text field variable to apply HTML tags to text fields.

The following HTML tags are supported by the `htmlText` property text fields: `a`, `b`, `font color`, `font face`, `font size`, `i`, `p`, and `u`.

The following HTML attributes are supported in text fields: `leftmargin`, `rightmargin`, `align`, `indent`, and `leading`. To apply these attributes, use the TextFormat class or cascading style sheets. For more information, see Chapter 9, "Working with Text," in *Using ActionScript in Flash* and "TextFormat class" or "TextField.StyleSheet class" in *Flash ActionScript Language Reference*.

To use the text field instance name to preserve rich text formatting:

1. Do one of the following to assign an instance name to the text field:
   - Use the Text tool to create a text field on the Stage. Assign the text field an instance name in the Property inspector.
   - Use the ActionScript `createTextField` method to create a text field dynamically. Assign the text field an instance name as a parameter of the `createTextField` method.
2. Do one of the following:
   - Select the Render Text as HTML option in the Property inspector.
   - In the Actions panel, set the `html` property of the TextField object to `true`, as in the following:

     ```
     instanceName.html = true;
     ```
3. In the Actions panel, set the `htmlText` property to a value that includes HTML tags.

   For example, if you have a dynamic text field on the Stage with the instance name `instName`, the following code renders the text in bold:

   ```
   instName.htmlText = "<b>Chris</b>";
   ```

To use the text field variable to preserve rich text formatting:

1. Select a text field on the Stage.
2. Assign the text field a variable name in the Property inspector.
3. Do one of the following:
   - Select the Render Text as HTML option in the Property inspector.
   - In the Actions panel, set the `html` property of the TextField object to `true`.

4. Set the text field variable to a value that includes HTML tags.

   For example, the following code assigns a value to a text field with the variable name `txt`. The text is rendered in bold if you select the Render Text as HTML option in the Property inspector, or if the `html` property is set to `true`:

   ```
   txt = "<b>Chris</b>";
   ```

   In the following example, the variable name of the text field is also `txt`. Because the value of the `html` property of the TextField object is set to `true`, you can use the variable name to render the text field in bold without selecting the Render Text as HTML option in the Property inspector:

   ```
   instName.html = true;
   txt = "<b>Chris</b>";
   ```

Substituting missing fonts

If you work with a document containing fonts that aren't installed on your system (for example, a document you received from another designer), Flash substitutes the missing fonts with fonts available on your system. You can select which fonts on your system are substituted for the missing fonts, or you can let Flash substitute missing fonts with the Flash System Default Font (specified in General Preferences).

Note: Substituting missing fonts while editing a Flash document does not change the fonts that are specified in the Flash document.

If you install a previously missing font on your system and restart Flash, the font is displayed in any documents using the font, and the font is removed from the Missing Fonts dialog box.

Selecting substitute fonts

An alert box indicating missing fonts in a document appears the first time a scene containing a missing font is displayed on the Stage. If you publish or export the document without displaying any scenes containing the missing fonts, the alert box appears during the publish or export operation. If you choose to select substitute fonts, the Font Mapping dialog box appears, listing all missing fonts in the document and letting you select a substitute for each.

Note: If the document contains many missing fonts, a delay may occur while Flash generates the list of missing fonts.

You can apply the missing font to new or existing text in the current document. The text is displayed on your system using the substitute font, but the missing font information is saved with the document. If the document is reopened on a system that includes the missing font, the text is displayed in that font.

Text attributes such as font size, leading, kerning, and so on may need to be adjusted when the text is displayed in the missing font, because the formatting you apply is based on the appearance of the text in the substitute font.

To specify font substitution:

1. Specify a font substitution preference. When the Missing Fonts alert appears, do one of the following:
   - Click Select Substitute Fonts to select substitute fonts from fonts installed on your system and proceed to step 2.
   - Click Use Default to use the Flash System Default Font to substitute all missing fonts and to dismiss the Missing Fonts alert.
2. In the Font Mapping dialog box, click a font in the Missing Fonts column to select it. Shift-click to select multiple missing fonts, to map them all to the same substitute font.

   The default substitute fonts are displayed in the Mapped To column, until you select substitute fonts.
3. Select a font from the Substitute Font pop-up menu.
4. Repeat steps 2–3 for all missing fonts.
5. Click OK.

Working with substitute fonts

You can use the Font Mapping dialog box to change the substitute font mapped to a missing font, to view all the substitute fonts you have mapped in Flash on your system, and to delete a substitute font mapping from your system. You can also turn off the Missing Fonts alert to prevent the alert from appearing.

When you work with a document that includes missing fonts, the missing fonts are displayed in the font list in the Property inspector. When you select substitute fonts, the substitute fonts are also displayed in the font list.

To view all the missing fonts in a document and reselect substitute fonts:

1. With the document active in Flash, select Edit > Font Mapping.
2. Select a substitute font, as described in the preceding procedure.

To view all the font mappings saved on your system and delete font mappings:

1. Close all documents in Flash.
2. Select Edit > Font Mapping.
3. To delete a font mapping, select the mapping and press Delete.
4. Click OK.

To turn off the Missing Fonts alert, do one of the following:

- To turn the alert off for the current document, in the Missing Fonts alert box select Don't Show Again for This Document, Always Use Substitute Fonts. Select Edit > Font Mapping to view mapping information for the document again.
- To turn the alert off for all documents, select Edit > Preferences (Windows) or Flash > Preferences (Macintosh) and click the Warnings tab. Deselect Warn on Missing Font and click OK. Select the option again to turn alerts on.

Controlling text with ActionScript

A dynamic or input text field is an instance of the ActionScript TextField object. When you create a text field, you can assign it an instance name in the Property inspector. You can use the instance name in ActionScript statements to set, change, and format the text field and its content using the TextField andTextFormat objects.

The TextField object has the same properties as the MovieClip object, and has methods that let you set, select, and manipulate the text. The TextFormat object lets you set character and paragraph values for the text. You can use these ActionScript objects instead of the text Property inspector to control the settings of a text field.

You can use a text field's variable name or instance name to assign it text that contains HTML tags. Flash preserves the rich text formatting applied to the text field with ActionScript.

If you assign a variable to a text field, the text field displays the variable's value. You can use ActionScript to pass the variable to other parts of the Flash application, to a server-side application for storing in a database, and so on. You can also replace the value of the variable by reading it from a server-side application or by loading it from another part of the Flash application. For more information on using variables, see "About variables" in *Using ActionScript Flash*. For more information about connecting to external applications, see Chapter 11, "Working with External Data,"in *Using ActionScript Flash*.

Creating and removing text fields dynamically

You can use the `createTextField` method of the MovieClip object to create a new, empty text field as a child of the movie clip that calls the method. You can use the `removeTextField` method to remove a text field created with `createTextField`; this method will not work on a text field created manually on the Timeline.

When you create a text field, you can use the TextField object to set properties of the text field. If you don't set the properties, the new text field receives a set of default properties. The default properties of the new text field are as follows:

```
type = "dynamic"
border = false
background = false
password = false
multiline = false
html = false
embedFonts = false
variable = null
maxChars = null
```

After you create a text field, you can use the TextFormat object to format the text. You must create a new TextFormat object and then pass it as a parameter to the `setTextFormat` method of the TextField object. A text field created with the `createTextField` method receives the following default TextFormat object:

```
font = "Times New Roman"
size = 12
color = 0x000000
bold = false
```

```
italic = false
underline = false
url = ""
target = ""
align = "left"
leftMargin = 0
rightMargin = 0
indent = 0
leading = 0
bullet = false
tabStops = [] (empty array)
```

To create a dynamic text field:

1. Select a frame, button, or movie clip that will receive the action.
2. Select Window > Development Panels > Actions to open the Actions panel if it isn't already open.
3. In the Actions toolbox, select the Built-in Classes category, then select the Movie category, then select the MovieClip category, and then select the Methods category. Finally, double-click the `createTextField` method.
4. Select the placeholder `instanceName` and enter an instance name or path for the movie clip that will be the parent of the new text field. For this example, enter the alias **_root** because the main Timeline is the parent.
5. Enter values for the following parameters:
   - *`Instance Name`* is the instance name of the new text field. For this example, enter **myText**.
   - *`Depth`* is a number that specifies the stacking order. For this example, enter **1**.
   - *`X`* is the *x* coordinate relative to the parent clip. For this example, enter **50**.
   - *`Y`* is the *y* coordinate relative to the parent clip. For this example, enter **50**.

   The following code is displayed in the Script pane:

   ```
   _root.createTextField("mytext",1,50,50,200,100);
   ```
6. In the Actions toolbox, select the Built-in Classes category, then select the Movie category, then select the TextField category, and then select the Properties category. Finally, double-click the `text` property to create a new line. For this example, replace the placeholder `instanceName` with **myText** in the Object parameter field.
7. In the Value field, enter **this is my first text field object text**. The following text is displayed in the Script pane:

   ```
   mytext.text = "this is my first text field object text";
   ```

   This example creates a text field with an instance name `myText`, a depth of 1, a width of 200, a height of 100, an *x* value of 50, and a *y* value of 50.

For a detailed description of the `createTextField` method of the TextField object, see "TextField class" in *Flash ActionScript Language Reference*.

Setting text field properties dynamically

To use ActionScript to set the properties of a text field, you must assign the text field an instance name. If you create the text field on the Stage with the Text tool, you can assign the instance name in the Property inspector. If you create the text field dynamically, you can assign an instance name as a parameter of the `createTextField` method.

To set text field properties dynamically:

1. Select Window > Development Panels > Actions to open the Actions panel if it isn't already open.
2. Do one of the following to create a text field:
   - Select the Text tool and create a text field on the Stage. Assign the text field an instance name in the Property inspector. For this example, enter the instance name **myText**.
   - Double-click the `createTextField` method of the MovieClip object in the Actions toolbox to add it to the Script pane in the Actions panel. See "Creating text" on page 108. For this example, enter the instance name **myText** as a parameter of the `createTextField` method.
3. Do one of the following to place text in the text field:
   - Enter text into the text field on the Stage.
   - Set the `text` property of the TextField object. See "Creating text" on page 108.
4. In the Actions toolbox, select the Built-in Classes category, then select the Movie category, then select the TextField category, and then select the Properties category. Finally, double-click the multiline property.
5. Enter the following parameters:
   - *`Object`* is the instance name of the text field whose property you want to set.
   - *`Value`* is the value of the property.
6. Repeat step 4 for the `wordWrap` and `border` properties. The following code appears in the Script pane:

```
mytext.multiline = true;
mytext.wordWrap = true;
mytext.border = true;
```

For a complete list of TextField object methods and detailed descriptions of each, see "TextField class" in *Flash ActionScript Language Reference*.

Formatting text dynamically

You can use the ActionScript TextFormat object to set properties of a text field. A TextFormat object incorporates character and paragraph formatting information. Character formatting information describes the appearance of individual characters: font name, point size, color, and an associated URL. Paragraph formatting information describes the appearance of a paragraph: left margin, right margin, indention of the first line, and left, right, or center alignment.

First you must create a new TextFormat object. Then you can use the methods of the TextField object and pass them the TextFormat object as a parameter to format the text in a field.

Each character in a text field may individually be assigned a TextFormat object. The TextFormat object of the first character of a paragraph is examined to perform paragraph formatting for the entire paragraph.

To format text dynamically:

1. Select Window > Development Panels > Actions to open the Actions panel if it isn't already open.
2. Do one of the following to create a text field:
   - Use the Text tool to create a text field on the Stage. Assign the text field an instance name in the Property inspector.
   - For this example, enter the instance name **myText**.
   - Use the `createTextField` method of the MovieClip object. See "Creating text" on page 108. For this example, enter the instance name **myText** as a parameter of the `createTextField` method.
3. Do one of the following to place text in the text field:
   - Enter text into the text field on the Stage.
   - Set the `text` property of the TextField object. See "Creating text" on page 108.
   - In the Actions toolbox, select the Built-in classes category, then select the Movie category, and then select the TextFormat category. Finally, double-click `new TextFormat`. For this example, enter **myformat** in the Object parameter field.

     The following code is displayed in the Script pane:

     ```
     myformat = new TextFormat();
     ```
4. In the Actions toolbox, select the Built-in Classes category, then select the Movie category, then select the TextFormat category, and then select the Properties category. Finally, double-click `color`. Repeat this step for the `bullet` and `underline` properties. The following code is displayed in the Script pane:

   ```
   myformat.color = 0xff0000;
   myformat.bullet = true;
   myformat.underline = true;
   ```
5. In the Actions toolbox, select the Built-in Classes category, then select the Movie category, then select the TextField category, and then select the Method category. Finally, double-click `setTextFormat`. For this example, enter **myText** in the Object parameter field.
6. In the Object field, enter the name of the TextFormat object you created in step 3, **myformat**. The following code appears in the Script pane:

   ```
   mytext.setTextFormat(myformat);
   ```

For more information, see "Using the TextFormat class" in *Using ActionScript in Flash*.

Using text field events to trigger scripts

You can use ActionScript to capture events that happen to text fields—for example, you can determine whether a user has changed or scrolled the text. You can write ActionScript statements that use these events to trigger scripts to run.

You can capture the following text field events: `onChanged` and `onScroller`.

To use a text field event to trigger a script:

1. Assign an instance name to the text field. Do one of the following:
   - Use the Text tool to create a text field on the Stage. Assign the text field an instance name in the Property inspector.
   - Use ActionScript to create a text field dynamically with the `createTextField` method. Assign the text field an instance name as a parameter of the `createTextField` method.
2. In the Actions panel, select the Built-in Classes category in the Actions toolbox, then select the Movie category, then select the TextField category, and then select the Events category. Finally, double-click an event. For this example, use the `onChanged` method.
3. Replace the placeholder `instanceName` with the actual instance name of the text field .
4. Add ActionScript statements inside the function. These statements run when the text field is changed.

About using Cascading Style Sheets (CSS) with text fields

You can attach style sheets to text fields to control text formatting. Flash supports a subset of CSS tags. You attach a style sheet to a text file using the TextField.StyleSheet object. See "Creating a style sheet object" in *Using ActionScript in Flash*.

Creating scrolling text

You can use the `scroll` and `maxscroll` properties of the TextField object to control vertical scrolling and the `hscroll` and `maxhscroll` properties to control horizontal scrolling in a text block. The `scroll` and `hscroll` properties contain a number that specifies the topmost visible line in a text block; you can read and write these properties. The `maxscroll` and `maxhscroll` properties contain a number that specifies the topmost visible line in a text block when the bottom line of the text is visible in the text block; you can only read these properties.

To use the scroll property to create scrolling text:

1. Assign an instance name to the text field that will contain scrolling text. Do one of the following:
   - Use the Text tool to create a text field on the Stage. Assign the text field an instance name in the Property inspector.
   - Use ActionScript to create a text field dynamically with the `createTextField` method. Assign the text field an instance name as a parameter of the `createTextField` method.
2. Create an Up button and a Down button or select Window > Other Panels > Common Libraries > Buttons and drag buttons to the Stage. You will use these buttons to scroll the text up and down.
3. Select the Up button on the Stage.
4. In the Actions panel, select the Built-in Classes category, then select the Movie category, then select the TextField category, and then select the Properties category. Finally, double-click the `scroll` property to add it to the Script pane.

5. Replace `instanceName` with the instance name of the text field you want to scroll.
6. Increment the scroll property by 1 to scroll the text up. The code should look like this:

```
instName.scroll += 1;
```

7. Select the Down button on the Stage.
8. Repeat steps 4 and 5.
9. Decrement the scroll property by 1 to scroll the text down. The code should look like this:

```
instName.scroll -= 1;
```

CHAPTER 7
Using Imported Artwork

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 can use artwork created in other applications. You can import vector graphics and bitmaps in a variety of file formats. If you have QuickTime 4 or later installed on your system, you can import additional vector or bitmap file formats. For more information, see "Importing file formats for vector or bitmap files" on page 133. You can import Macromedia FreeHand files (version MX and earlier) and Macromedia Fireworks PNG files directly into Flash, preserving attributes from those formats.

When you import a bitmap, you can apply compression and anti-aliasing, place the bitmap directly in a Flash document, use the bitmap as a fill, edit the bitmap in an external editor, break the bitmap apart into pixels and edit it in Flash, or convert the bitmap to vector artwork. See "Working with imported bitmaps" on page 138.

You can also import video into Flash. See Chapter 10, "Working with Video," on page 177.

For information on importing sound files in WAV (Windows), AIFF (Macintosh), and MP3 (both platforms) formats, see Chapter 11, "Working with Sound," on page 201.

Placing artwork into Flash

Flash recognizes a variety of vector and bitmap formats. You can place artwork into Flash by importing it onto the Stage in the current Flash document or into the library for the current document. You can also import bitmaps by pasting them on the Stage in the current document. All bitmaps that you import directly into a Flash document are automatically added to the document's library.

Graphic files that you import into Flash must be at least 2 pixels x 2 pixels in size.

You can load JPEG files into a Flash movie during runtime using the `loadMovie` action or method. For detailed information, see `loadMovie()` in Flash ActionScript Language Reference.

Flash imports vector graphics, bitmaps, and sequences of images as follows:

- When you import vector images into Flash from FreeHand, you can select options for preserving FreeHand layers, pages, and text blocks. See "Importing FreeHand MX files" on page 135.

- When you import PNG images from Fireworks, you can import files as editable objects that you can modify in Flash, or as flattened files that you can edit and update in Fireworks.
- You can select options for preserving images, text, and guides. See "Importing Fireworks PNG files" on page 134.

  Note: If you import a PNG file from Fireworks by cutting and pasting, the file is converted to a bitmap.

- When you import Adobe Illustrator, EPS, or PDF files into Flash, you can select options for converting pages and layers. You can choose to rasterize all content, including text. See "Importing Adobe Illustrator, EPS, or PDF files" on page 136.
- Vector images from SWF and Windows Metafile Format (WMF) files that you import directly into a Flash document (instead of into a library) are imported as a group in the current layer. See "Importing file formats for vector or bitmap files" on page 133 and "Importing Adobe Illustrator, EPS, or PDF files" on page 136.
- Bitmaps (scanned photographs, BMP files) that you import directly into a Flash document are imported as single objects in the current layer. Flash preserves the transparency settings of imported bitmaps. Because importing a bitmap can increase the file size of a SWF file, you may want to compress imported bitmaps. See "Setting bitmap properties" on page 139.

  Note: Bitmap transparency may not be preserved when bitmaps are imported by dragging and dropping from an application or desktop to Flash. To preserve transparency, use the File > Import to Stage or Import to Library command for importing.

- Any sequence of images (for example, a PICT and BMP sequence) that you import directly into a Flash document is imported as successive keyframes of the current layer.

For information on specific file formats, see "Importing file formats for vector or bitmap files" on page 133.

To import a file into Flash:

1. Do one of the following:
   - To import a file directly into the current Flash document, select File > Import to Stage.
   - To import a file into the library for the current Flash document, select File > Import to Library. (To use a library item in a document, drag it onto the Stage. See Chapter 3, "Using Symbols, Instances, and Library Assets," on page 53.)
2. In the Import dialog box, select a file format from the Files of Type (Windows) or Show (Macintosh) pop-up menu.
3. Navigate to the desired file and select it.

   If an imported file has multiple layers, Flash may create new layers (depending on the import file type). Any new layers will be displayed in the Timeline.

   Note: If you are importing a Fireworks PNG file, see "Importing Fireworks PNG files" on page 134. If you are importing a FreeHand file, see "Importing FreeHand MX files" on page 135. If you are importing an Adobe Illustrator file, see "Importing Adobe Illustrator, EPS, or PDF files" on page 136.

4. Click Open.

5. If the name of the file you are importing ends with a number, and there are additional sequentially numbered files in the same folder, select whether to import the sequence of files.
   - Click Yes to import all the sequential files.
   - Click No to import only the specified file.

   The following are examples of filenames that can be used as a sequence:

   Frame001.gif, Frame002.gif, Frame003.gif

   Bird 1, Bird 2, Bird 3

   Walk-001.ai, Walk-002.ai, Walk-003.ai

To paste a bitmap from another application directly into the current Flash document:

1. Copy the image in the other application.
2. In Flash, select Edit > Paste in Center or Edit > Paste in Place.

Importing file formats for vector or bitmap files

Flash can import different vector or bitmap file formats depending on whether QuickTime 4 or later is installed on your system. Using Flash with QuickTime 4 installed is especially useful for collaborative projects in which authors work on both Windows and Macintosh platforms. QuickTime 4 extends support for certain file formats (including Adobe Photoshop, PICT, QuickTime Movie, and others) to both platforms.

The following vector or bitmap file formats can be imported into Flash MX 2004, regardless of whether QuickTime 4 is installed:

File type	Extension	Windows	Macintosh
Adobe Illustrator (version 10 or earlier; see "Importing Adobe Illustrator, EPS, or PDF files" on page 136)	.eps, .ai .pdf	✔	✔
AutoCAD DXF (see "AutoCAD DXF files" on page 137)	.dxf	✔	✔
Bitmap	.bmp	✔	✔ (Using QuickTime)
Enhanced Windows Metafile	.emf	✔	
FreeHand	.fh7, .fh8, .fh9, .fh10, .fh11	✔	✔
FutureSplash Player	.spl	✔	✔
GIF and animated GIF	.gif	✔	✔
JPEG	.jpg	✔	✔
PNG	.png	✔	✔
Flash Player 6/7	.swf	✔	✔
Windows Metafile	.wmf	✔	✔

The following bitmap file formats can be imported into Flash only if QuickTime 4 or later is installed:

File type	Extension	Windows	Macintosh
MacPaint	.pntg	✔	✔
Photoshop	.psd	✔	✔
PICT	.pct, .pic	✔ (As bitmap)	✔
QuickTime Image	.qtif	✔	✔
Silicon Graphics Image	.sgi	✔	✔
TGA	.tga	✔	✔
TIFF	.tif	✔	✔

Importing Fireworks PNG files

You can import Fireworks PNG files into Flash as flattened images or as editable objects. When you import a PNG file as a flattened image, the entire file (including any vector artwork) is *rasterized,* or converted to a bitmap image. When you import a PNG file as editable objects, vector artwork in the file is preserved in vector format. You can choose to preserve placed bitmaps, text, and guides in the PNG file when you import it as editable objects.

If you import the PNG file as a flattened image, you can start Fireworks from within Flash and edit the original PNG file (with vector data). See "Editing bitmaps in an external editor" on page 140.

When you import multiple PNG files in a batch, you select import settings one time. Flash uses the same settings for all files in the batch.

Note: You can edit bitmap images in Flash by converting the bitmap images to vector artwork or by breaking apart the bitmap images. See "Converting bitmaps to vector graphics" on page 142 and "Breaking apart a bitmap" on page 141.

To import a Fireworks PNG file:

1. Select File > Import to Stage or Import to Library.
2. In the Import dialog box, select PNG Image from the Files of Type (Windows) or Show (Macintosh) pop-up menu.
3. Navigate to a Fireworks PNG image and select it.
4. Click Open.
5. In the Fireworks PNG Import Settings dialog box, select one of the following for File Structure:

   Import as Movie Clip and Retain Layers imports the PNG file as a movie clip, with all of its frames and layers intact inside the movie clip symbol.

   Import into New Layer in Current Scene imports the PNG file into the current Flash document in a single new layer at the top of the stacking order. The Fireworks layers are flattened into the single layer. The Fireworks frames are contained in the new layer.

6. For Objects, select one of the following:

   Rasterize if Necessary to Maintain Appearance preserves Fireworks fills, strokes, and effects in Flash.

   Keep All Paths Editable keeps all objects as editable vector paths. Some Fireworks fills, strokes, and effects are lost on import.

7. For Text, select one of the following:

   Rasterize if Necessary to Maintain Appearance preserves Fireworks fills, strokes, and effects in text imported into Flash.

   Keep All Paths Editable keeps all text editable. Some Fireworks fills, strokes, and effects are lost on import.

8. Select Import as a Single Flattened Image to flatten the PNG file into a single bitmap image. When this option is selected, all other options are dimmed.

9. Click OK.

Importing FreeHand MX files

You can import FreeHand files in version 7 or later directly into Flash. FreeHand MX is the best choice for creating vector graphics for import into Flash, because you can preserve FreeHand layers, text blocks, library symbols, and pages, and choose a page range to import. If the imported FreeHand file is in CMYK color mode, Flash converts the file to RGB.

Keep the following guidelines in mind when importing FreeHand files:

- When importing a file with overlapping objects that you want to preserve as separate objects, place the objects on separate layers in FreeHand, and select Layers in the FreeHand Import dialog box in Flash when importing the file. (If overlapping objects on a single layer are imported into Flash, the overlapping shapes will be divided at intersection points, just as with overlapping objects that you create in Flash.)
- When you import files with gradient fills, Flash can support up to eight colors in a gradient fill. If a FreeHand file contains a gradient fill with more than eight colors, Flash creates clipping paths to simulate the appearance of a gradient fill. Clipping paths can increase file size. To minimize file size, use gradient fills with eight colors or fewer in FreeHand.
- When you import files with blends, Flash imports each step in a blend as a separate path. Thus, the more steps a blend has in a FreeHand file, the larger the imported file size will be in Flash.
- When you import files with strokes that have square caps, Flash converts the caps to round caps.
- When you import files with placed grayscale images, Flash converts the grayscale images to RGB images. This conversion can increase the imported file's size.
- When importing files with placed EPS images, you must select the Convert Editable EPS when Imported option in FreeHand Import Preferences before you place the EPS into FreeHand. If you do not select this option, the EPS image will not be viewable when imported into Flash. In addition, Flash does not display information for an imported EPS image (regardless of the Preferences settings used in FreeHand).

To import a FreeHand file:

1. Select File > Import to Stage or File > Import to Library.
2. In the Import dialog box, select FreeHand from the Files of Type (Windows) or Show (Macintosh) pop-up menu.
3. Navigate to a FreeHand file and select it.
4. Click Open.
5. In the FreeHand Import Settings dialog box, for Mapping Pages, select a setting:

   Scenes converts each page in the FreeHand document to a scene in the Flash document.

   Keyframes converts each page in the FreeHand document to a keyframe in the Flash document.
6. For Mapping Layers, select one of the following:

   Layers converts each layer in the FreeHand document to a layer in the Flash document.

   Keyframes converts each layer in the FreeHand document to a keyframe in the Flash document.

   Flatten converts all layers in the FreeHand document to a single flattened layer in the Flash document.
7. For Pages, do one of the following:
   - Select All to import all pages from the FreeHand document.
   - Enter page numbers for From and To to import a page range from the FreeHand document.
8. For Options, select any of the following options:

   Include Invisible Layers imports all layers (visible and hidden) from the FreeHand document.

   Include Background Layer imports the background layer with the FreeHand document.

   Maintain Text Blocks preserves text in the FreeHand document as editable text in the Flash document.
9. Click OK.

Importing Adobe Illustrator, EPS, or PDF files

Flash can import Adobe Illustrator files in version 10 or earlier, EPS files in any version, and PDF files in version 1.4 or earlier.

Note: The PDF version number is different from the Adobe Acrobat number. Adobe Acrobat is a product used to author PDF files. PDF is the file format.

When you import an Illustrator file into Flash, you must ungroup all the Illustrator objects on all layers. Once all the objects are ungrouped, they can be manipulated like any other Flash object. You can also export Flash documents as Adobe Illustrator files. For information on exporting Illustrator files, see "Adobe Illustrator" on page 347.

You can choose from the following options when importing Adobe Illustrator, EPS, or PDF files:

- Convert pages to scenes or keyframes.
- Convert layers to Flash layers or keyframes or flatten all layers.

- Select which pages to import.
- Include invisible layers.
- Maintain text blocks.
- Rasterize everything. Choosing this option flattens layers and rasterizes text, and disables options for converting layers or maintaining text blocks.

To import an Adobe Illustrator, EPS, or PDF file:

1. Select File > Import to Stage or Import to Library.
2. In the Import dialog box, select Adobe Illustrator, EPS, or PDF from the Files of Type (Windows) or Show (Macintosh) pop-up menu.
3. Navigate to a file and select it.
4. Click Open.

   The Import Options dialog box appears.
5. For Convert Pages, select one of the following:

   Screens (in screens mode) or **Scenes** (in scenes mode) converts each page to a screen or a scene.

   Keyframes converts each page to a keyframe.
6. For Convert Layers, select one of the following:

   Layers converts each layer in the imported document to a layer in the Flash document.

   Keyframes converts each layer in the imported document to a keyframe in the Flash document.

   Flatten converts all layers in the imported document to a single flattened layer in the Flash document.
7. For Which Pages to Import, select All to import all pages, or select From and enter a page range to import.
8. For Options, select any of the following:

   Include Invisible Layers imports all layers (visible and hidden) from the imported document.

   Maintain Text Blocks imports text as editable text in Flash.

   Rasterize Everything converts all content in the imported document to bitmaps. Enter a value to set the resolution for the imported document. Selecting this option flattens all layers and disables the Maintain Text Blocks option.
9. Click OK.

AutoCAD DXF files

Flash supports the AutoCAD DXF format in the release 10 version.

DXF files do not support the standard system fonts. Flash tries to map fonts appropriately, but the results can be unpredictable, particularly for the alignment of text.

Since the DXF format does not support solid fills, filled areas are exported as outlines only. For this reason, the DXF format is most appropriate for line drawings, such as floor plans and maps.

You can import two-dimensional DXF files into Flash. Flash does not support three-dimensional DXF files.

Although Flash doesn't support scaling in a DXF file, all imported DXF files produce 12-inch x 12-inch files that you can scale using the Modify > Transform > Scale command. Also, Flash supports only ASCII DXF files. If your DXF files are binary, you must convert them to ASCII before importing them into Flash.

Working with imported bitmaps

When you import a bitmap into Flash, you can modify that bitmap and use it in your Flash document in a variety of ways. You can apply compression and anti-aliasing to imported bitmaps to control the size and appearance of bitmaps in your Flash applications. See "Setting bitmap properties" on page 139. You can apply an imported bitmap as a fill to an object. See "Applying a bitmap fill" on page 140.

Flash lets you break apart a bitmap into editable pixels. The bitmap retains its original detail but is broken into discrete areas of color. When you break a bitmap apart, you can select and modify areas of the bitmap with the Flash drawing and painting tools. Breaking apart a bitmap also lets you sample the bitmap with the Eyedropper tool to use it as a fill. See "Breaking apart a bitmap" on page 141.

You can edit an imported bitmap in Fireworks or another external image editor by starting the editing application from within Flash. See "Editing bitmaps in an external editor" on page 140. To convert a bitmap's image to a vector graphic, you can trace the bitmap. Performing this conversion enables you to modify the graphic as you do other vector artwork in Flash. See "Converting bitmaps to vector graphics" on page 142.

If a Flash document displays an imported bitmap at a larger size than the original, the image may be distorted. Preview imported bitmaps to be sure that images are displayed properly.

Using the Property inspector to work with bitmaps

When you select a bitmap on the Stage, the Property inspector displays the bitmap's symbol name and its pixel dimensions and position on the Stage. Using the Property inspector, you can assign a new name to the bitmap, and you can *swap* an instance of a bitmap—that is, replace the instance with an instance of another bitmap in the current document.

To display the Property inspector with bitmap properties:

1. Select an instance of a bitmap on the Stage.
2. Select Window > Properties.

To assign a new name to a bitmap:

1. Select the bitmap in the Library panel.
2. Select Window > Properties if the Property inspector is not visible. Select an instance of the bitmap on the Stage to view the bitmap properties.
3. In the Property inspector, enter a new name in the Name text box.
4. Click OK.

To replace an instance of a bitmap with an instance of another bitmap:

1. Select a bitmap instance on the Stage.
2. Select Window > Properties if the Property inspector is not visible.
3. In the Property inspector, click Swap.
4. In the Swap Bitmap dialog box, select a bitmap to replace the one currently assigned to the instance.

Setting bitmap properties

You can apply anti-aliasing to an imported bitmap to smooth the edges in the image. You can also select a compression option to reduce the bitmap file size and format the file for display on the web.

To select bitmap properties, you use the Bitmap Properties dialog box.

To set bitmap properties:

1. Select a bitmap in the Library panel.
2. Do one of the following:
   - Click the properties icon at the bottom of the Library panel.
   - Right-click (Windows) or Control-click (Macintosh) the bitmap's icon and select Properties from the context menu.
   - Select Properties from the options menu in the upper right corner of the Library panel.
3. In the Bitmap Properties dialog box, select Allow Smoothing to smooth the edges of the bitmap with anti-aliasing.
4. For Compression, select one of the following options:

   Photo (JPEG) compresses the image in JPEG format. To use the default compression quality specified for the imported image, select Use Document Default Quality. To specify a new quality compression setting, deselect Use Document Default Quality and enter a value between 1 and 100 in the Quality text box. (A higher setting preserves greater image integrity but yields a larger file size.)

   Lossless (PNG/GIF) compresses the image with lossless compression, in which no data is discarded from the image.

   Note: Use Photo compression for images with complex color or tonal variations, such as photographs or images with gradient fills. Use Lossless compression for images with simple shapes and relatively few colors.

5. Click Test to determine the results of the file compression. Compare the original file size to the compressed file size to determine if the selected compression setting is acceptable.
6. Click OK.

   Note: JPEG Quality settings that you select in the Publish Settings dialog box do not specify a quality setting for imported JPEG files. You must specify a quality setting for imported JPEG files in the Bitmap Properties dialog box.

Applying a bitmap fill

You can apply a bitmap as a fill to a graphic object using the Color Mixer. Applying a bitmap as a fill tiles the bitmap to fill the object. The Fill Transform tool allows you to scale, rotate, or skew an image and its bitmap fill. See "Transforming gradient and bitmap fills" on page 80.

To apply a bitmap as a fill using the Color Mixer:

1. To apply the fill to existing artwork, select a graphic object or objects on the Stage.
2. Select Window > Design Panels > Color Mixer.
3. In the Color Mixer, select Bitmap from the pop-up menu in the center of the panel.
4. If you need a larger preview window to display more bitmaps in the current document, click the arrow in the lower right corner to expand the Color Mixer.
5. Click a bitmap to select it.

   The bitmap becomes the current fill color. If you selected artwork in step 1, the bitmap is applied as a fill to the artwork.

Editing bitmaps in an external editor

If you are editing a Fireworks PNG file imported as a flattened image, you can choose to edit the PNG source file for the bitmap, when available.

Note: You cannot edit bitmaps from Fireworks PNG files imported as editable objects in an external image editor.

If you have Fireworks 3 or later or another image-editing application installed on your system, you can start the application from within Flash to edit an imported bitmap.

To edit a bitmap with Fireworks 3 or later:

1. In the Library panel, right-click (Windows) or Control-click (Macintosh) the bitmap's icon.
2. In the bitmap's context menu, select Edit with Fireworks 3.
3. In the Edit Image dialog box, specify whether the PNG source file or the bitmap file is to be opened.
4. Perform the desired modifications to the file in Fireworks.
5. In Fireworks, select File > Update.
6. Return to Flash.

   The file is automatically updated in Flash.

To edit a bitmap with another image-editing application:

1. In the Library panel, right-click (Windows) or Control-click (Macintosh) the bitmap's icon.
2. In the bitmap's context menu, select Edit With.
3. Select an image-editing application to open the bitmap file, and click OK.
4. Perform the desired modifications to the file in the image-editing application.

5. Save the file in the image-editing application.

   The file is automatically updated in Flash.

6. Return to Flash to continue editing the document.

Breaking apart a bitmap

Breaking apart a bitmap separates the pixels in the image into discrete areas that can be selected and modified separately. When you break apart a bitmap, you can modify the bitmap with the Flash drawing and painting tools. Using the Lasso tool with the Magic Wand modifier, you can select areas of a bitmap that has been broken apart.

You can paint with a broken-apart bitmap by selecting the bitmap with the Eyedropper tool and applying the bitmap as a fill with the Paint Bucket tool or another drawing tool.

To break apart a bitmap:

1. Select a bitmap in the current scene.
2. Select Modify > Break Apart.

To change the fill of selected areas of a broken-apart bitmap:

1. Select the Lasso tool and click the Magic Wand modifier.

2. Click the Magic Wand Settings modifier and set the following options:
   - For Threshold, enter a value between 1 and 200 to define how closely the color of adjacent pixels must match to be included in the selection. A higher number includes a broader range of colors. If you enter 0, only pixels of the exact same color as the first pixel you click are selected.
   - For Smoothing, select an option from the pop-up menu to define how much the edges of the selection will be smoothed.
3. Click the bitmap to select an area. Continue clicking to add to the selection.
4. Select the fill that you want to use to fill the selected areas in the bitmap. See "Using the Stroke Color and Fill Color controls in the Tools panel" on page 76.
5. Select the Paint Bucket tool and click anywhere in the selected area to apply the new fill.

To apply a broken-apart bitmap as a fill using the Eyedropper tool:

1. Select the Eyedropper tool and click the broken-apart bitmap on the Stage.

   The Eyedropper tool sets the bitmap to be the current fill and changes the active tool to the Paint Bucket.

2. Do one of the following:
   - Click an existing graphic object with the Paint Bucket tool to apply the bitmap as a fill.
   - Select the Oval, Rectangle, or Pen tool and draw a new object. The object is filled with the broken-apart bitmap.

   You can use the Paint Bucket tool to scale, rotate, or skew the bitmap fill.

Converting bitmaps to vector graphics

The Trace Bitmap command converts a bitmap into a vector graphic with editable, discrete areas of color. This command lets you manipulate the image as a vector graphic; it is also useful if you want to reduce file size.

When you convert a bitmap to a vector graphic, the vector graphic is no longer linked to the bitmap symbol in the Library panel.

Note: If the imported bitmap contains complex shapes and many colors, the converted vector graphic may have a larger file size than the original bitmap. Try a variety of settings in the Trace Bitmap dialog box to find a balance between file size and image quality.

You can also break apart a bitmap in order to modify the image using Flash drawing and painting tools. See "Breaking apart a bitmap" on page 141.

To convert a bitmap to a vector graphic:

1. Select a bitmap in the current scene.
2. Select Modify > Bitmap > Trace Bitmap.
3. Enter a Color Threshold value between 1 and 500.

   When two pixels are compared, if the difference in the RGB color values is less than the color threshold, the two pixels are considered the same color. As you increase the threshold value, you decrease the number of colors.
4. For Minimum Area, enter a value between 1 and 1000 to set the number of surrounding pixels to consider when assigning a color to a pixel.
5. For Curve Fit, select an option from the pop-up menu to determine how smoothly outlines are drawn.
6. For Corner Threshold, select an option from the pop-up menu to determine whether sharp edges are retained or smoothed out.

   To create a vector graphic that looks most like the original bitmap, enter the following values:
   - Color Threshold: 10
   - Minimum Area: 1 pixel
   - Curve Fit: Pixels
   - Corner Threshold: Many Corners

CHAPTER 8
Working with Graphic Objects

In Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004, graphic objects are items on the Stage. Flash lets you move, copy, delete, transform, stack, align, and group graphic objects. You can also link a graphic object to a URL. Keep in mind that modifying lines and shapes can alter other lines and shapes on the same layer. See Chapter 5, "Drawing," on page 85.

Note: Graphic objects in Flash are different from ActionScript objects, which are part of the ActionScript programming language. Be careful not to confuse the two uses of the term. For more information on objects in the programming language, see "About data types" in *Using ActionScript in Flash*.

This chapter contains the following sections:

Selecting objects

To modify an object, you must first select it. Flash provides a variety of methods for making selections, including the Selection tool, the Lasso tool, and keyboard commands. You can group individual objects to manipulate them as a single object. See "Grouping objects" on page 146.

Flash highlights objects and strokes that have been selected with a dot pattern. Selected groups are highlighted with bounding boxes in the color used for the outline of the layer that contains the selected group. You can change the layer outline color in the Layer Properties dialog box. See "Using layers" in *Getting Started with Flash*.

You can choose to select only an object's strokes or only its fills. You can hide selection highlighting in order to edit objects without viewing highlighting.

When you select an object, the Property inspector displays the object's stroke and fill, its pixel dimensions, and the *x* and *y* coordinates of the object's transformation point.

If you select multiple items of different types on the Stage, such as an object, a button, and a movie clip, the Property inspector indicates a mixed selection. The Property inspector for a mixed selection displays the pixel dimensions and *x* and *y* coordinates of the selected set of items.

You can use the Property inspector for a shape to change the object's stroke and fill. See Chapter 4, "Working with Color," on page 75.

You might want to prevent a group or symbol from being selected and accidentally changed. To do this, you can lock the group or symbol. See "Modifying selections" on page 145.

Selecting objects with the Selection tool

The Selection tool lets you select entire objects by clicking an object or dragging to enclose the object within a rectangular selection marquee.

Note: To select the Selection tool, you can also press the V key. To temporarily switch to the Selection tool when another tool is active, hold down the Control key (Windows) or Command key (Macintosh).

To select a stroke, fill, group, instance, or text block:

- Select the Selection tool and click the object.

To select connected lines:

- Select the Selection tool and double-click one of the lines.

To select a filled shape and its stroked outline:

- Select the Selection tool and double-click the fill.

To select objects within a rectangular area:

- Select the Selection tool and drag a marquee around the object or objects that you want to select. Instances, groups, and type blocks must be completely enclosed to be selected.

Modifying selections

You can add to selections, select or deselect everything on every layer in a scene, select everything between keyframes, or lock and unlock selected symbols or groups.

To add to a selection:

- Hold down the Shift key while making additional selections.

  Note: To disable the Shift-selecting option, deselect the option in Flash General Preferences. See "Setting preferences in Flash" in *Getting Started with Flash*.

To select everything on every layer of a scene:

- Select Edit > Select All, or press Control+A (Windows) or Command+A (Macintosh).

  Select All doesn't select objects on locked or hidden layers, or layers not on the current Timeline.

To deselect everything on every layer:

- Select Edit > Deselect All, or press Control+Shift+A (Windows) or Command+Shift+A (Macintosh).

To select everything on one layer between keyframes:

- Click a frame in the Timeline. For more information, see "Using the Timeline" in *Getting Started with Flash*.

To lock a group or symbol:

- Select the group or symbol and select Modify > Arrange > Lock.

  Select Modify > Arrange > Unlock All to unlock all locked groups and symbols.

Selecting objects with the Lasso tool

To select objects by drawing either a freehand or a straight-edged selection area, you can use the Lasso tool and its Polygon Mode modifier. When using the Lasso tool, you can switch between the freeform and straight-edged selection modes.

To select objects by drawing a freehand selection area:

- Select the Lasso tool and drag around the area. End the loop approximately where you started, or let Flash automatically close the loop with a straight line.

To select objects by drawing a straight-edged selection area:

1. Select the Lasso tool and select the Polygon Mode modifier in the Options section of the Tools panel.
2. Click to set the starting point.
3. Position the pointer where you want the first line to end, and click. Continue setting end points for additional line segments.
4. To close the selection area, double-click.

To select objects by drawing both freehand and straight-edged selection areas:

1. Select the Lasso tool and deselect the Polygon Mode modifier.
2. To draw a freehand segment, drag on the Stage.
3. To draw a straight-edged segment, Alt-click (Windows) or Option-click (Macintosh) to set start and end points. You can continue switching between drawing freehand and straight-edged segments.
4. To close the selection area, do one of the following:
   - If you are drawing a freehand segment, release the mouse button.
   - If you are drawing a straight-edged segment, double-click.

Hiding selection highlighting

You can hide selection highlights in order to edit objects without viewing their highlighting. Hiding highlights enables you to see how artwork will appear in its final state while you are selecting and editing objects.

To hide selection highlighting:

- Select View > Hide Edges.

  Select the command again to deselect the feature.

Grouping objects

To manipulate elements as a single object, you need to group them. For example, after creating a drawing such as a tree or flower, you might group the elements of the drawing so that you can easily select and move the drawing as a whole.

When you select a group, the Property inspector displays the *x* and *y* coordinates of the group and its pixel dimensions.

You can edit groups without ungrouping them. You can also select an individual object in a group for editing, without ungrouping the objects.

To create a group:

1. Select the objects on the Stage that you want to group.

   You can select shapes, other groups, symbols, text, and so on.
2. Select Modify > Group, or press Control+G (Windows) or Command+G (Macintosh).

To ungroup objects:

- Select Modify > Ungroup, or press Control+Shift+G (Windows) or Command+Shift+G (Macintosh).

To edit a group or an object within a group:

1. With the group selected, select Edit > Edit Selected, or double-click the group with the Selection tool.

   Everything on the page that is not part of the group is dimmed, indicating it is inaccessible.
2. Edit any element within the group.
3. Select Edit > Edit All, or double-click a blank spot on the Stage with the Selection tool.

   Flash restores the group to its status as a single entity, and you can work with other elements on the Stage.

Moving, copying, and deleting objects

You can move objects by dragging them on the Stage, cutting and pasting them, using the arrow keys, or using the Property inspector to specify an exact location for them. You can also move objects between Flash and other applications using the Clipboard. You can copy objects by dragging or pasting them, or while transforming them. When you move an object, the Property inspector indicates the new position.

When moving an object with the Selection tool, you can use the Snap modifier for the Selection tool to quickly align the object with points on other objects.

Moving objects

To move an object, you can drag the object, use the arrow keys, use the Property inspector, or use the Info panel.

To move objects by dragging:

1. Select an object or multiple objects.
2. Select the Selection tool, position the pointer over the object, and drag to the new position. To copy the object and move the copy, Alt-drag (Windows) or Option-drag (Macintosh). To constrain the object's movement to multiples of 45°, Shift-drag.

To move objects using the arrow keys:

1. Select an object or multiple objects.
2. Press the arrow key for the direction in which you want the object to move 1 pixel at a time. Press Shift+arrow key to move the selection 10 pixels at a time.

   Note: When Snap to Pixels is selected, the arrow keys move objects by pixel increments on the document's pixel grid, not by pixels on the screen. See "Pixel snapping" on page 101.

To move objects using the Property inspector:

1. Select an object or multiple objects.
2. If the Property inspector is not visible, select Window > Properties.
3. Enter *x* and *y* values for the location of the upper left corner of the selection. The units are relative to the upper left corner of the Stage.

   Note: The Property inspector uses the units specified for the Ruler Units option in the Document Properties dialog box. To change the units, see "Creating or opening a document and setting properties" on page 12.

To move objects using the Info panel:

1. Select an object or multiple objects.
2. If the Info Panel is not visible, select Window > Design Panels > Info.
3. Enter *x* and *y* values for the location of the upper left corner of the selection. The units are relative to the upper left corner of the Stage.

Moving and copying objects by pasting

When you need to move or copy objects between layers, scenes, or other Flash files, you should use the pasting technique. You can paste an object in a position relative to its original position.

To move or copy objects by pasting:

1. Select an object or multiple objects.
2. Select Edit > Cut or Edit > Copy.
3. Select another layer, scene, or file and select Edit > Paste in Place to paste the selection in the same position relative to the Stage.

About copying artwork with the Clipboard

Elements copied to the Clipboard are anti-aliased, so they look as good in other applications as they do in Flash. This is particularly useful for frames that include a bitmap image, gradients, transparency, or a mask layer.

Graphics pasted from other Flash documents or programs are placed in the current frame of the current layer. How a graphic element is pasted into a Flash scene depends on the type of element it is, its source, and the preferences you have set:

- Text from a text editor becomes a single text object.
- Vector-based graphics from any drawing program become a group that can be ungrouped and edited like any other Flash element.
- Bitmaps become a single grouped object just like imported bitmaps. You can break apart pasted bitmaps or convert pasted bitmaps to vector graphics.

  For information on converting bitmaps to vector graphics, see "Converting bitmaps to vector graphics" on page 142.

  Note: Before pasting graphics from FreeHand into Flash, set your FreeHand export preferences to convert colors to CMYK and RGB for Clipboard formats.

Copying transformed objects

To create a scaled, rotated, or skewed copy of an object, you can use the Transform panel.

To create a transformed copy of an object:

1. Select an object.
2. Select Window > Design Panels > Transform.

3. Enter scale, rotation, or skew values. See "Scaling objects" on page 153, "Rotating objects" on page 153, and "Skewing objects" on page 154.
4. Click the Create Copy button in the Transform panel (the left button in the lower right corner of the panel).

Deleting objects

Deleting an object removes it from the file. Deleting an instance on the Stage does not delete the symbol from the library.

To delete objects:

1. Select an object or multiple objects.
2. Do one of the following:
   - Press Delete or Backspace.
   - Select Edit > Clear.
   - Select Edit > Cut.
   - Right-click (Windows) or Control-click (Macintosh) the object and select Cut from the context menu.

Stacking objects

Within a layer, Flash stacks objects based on the order in which they were created, placing the most recently created object at the top of the stack. The stacking order of objects determines how they appear when they are overlapping.

Drawn lines and shapes always appear below groups and symbols on the stack. To move them up the stack, you must group them or make them into symbols. You can change the stacking order of objects at any time.

Layers also affect the stacking order. Everything on Layer 2 appears on top of everything on Layer 1, and so on. To change the order of layers, drag the layer name in the Timeline to a new position. See "Using layers" in *Getting Started with Flash*.

To change the stacking order of an object:

1. Select the object.
2. Use one of the following commands:
   - Select Modify > Arrange > Bring to Front or Send to Back to move the object or group to the top or bottom of the stacking order.
   - Select Modify > Arrange > Bring Forward or Send Backward to move the object or group up or down one position in the stacking order.

   If more than one group is selected, the groups move in front of or behind all unselected groups, while maintaining their order relative to each other.

Transforming objects

You can transform graphic objects, as well as groups, text blocks, and instances, by using the Free Transform tool or the options in the Modify > Transform submenu. Depending on the type of element you select, you can freely transform, rotate, skew, scale, or distort the element. You can change or add to a selection during a transformation operation.

When you transform an object, group, text box, or instance, the Property inspector for that item displays any changes made to the item's dimensions or position.

A bounding box is displayed during transform operations that involve dragging. The bounding box is rectangular (unless it has been modified with the Distort command or the Envelope modifier; see "Distorting objects" on page 152 and "Modifying shapes with the Envelope modifier" on page 152) with its edges initially aligned to the edges of the Stage. Transformation handles are located on each corner and in the middle of each side. As you drag, the bounding box previews the transformations.

Working with the center point during transformations

During a transformation, a transformation point appears at the center of a selected element. The transformation point is initially aligned with the object's center point. You can move the transformation point, and you can return it to its default location.

For scaling, skewing, or rotating graphic objects, groups, and text blocks, the point opposite the point you drag is the point of origin by default. For instances, the transformation point is the point of origin by default. You can move the default point of origin for a transformation.

You can track the location of the transformation point in the Info panel, and in the Property inspector for the graphic object.

To move the transformation point during a transform operation:

- Drag the transformation point.

To realign the transformation point with the element's center point:

- Double-click the transformation point.

To switch the point of origin for a scale or skew transformation:

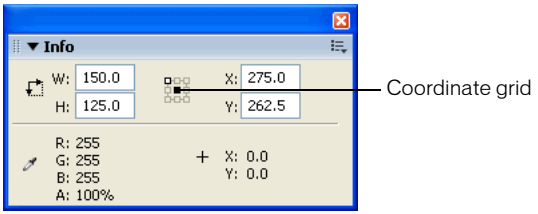
- Alt-drag (Windows) or Option-drag (Macintosh) during the transformation.

To track the location of the transformation point in the Info panel and Property inspector:

- In the Info panel, click the center square in the coordinate grid to select it.

  The selected square becomes black.

When the center square is selected, the X and Y values to the right of the coordinate grid in the Info panel display the *x* and *y* coordinates of the transformation point. In addition, the X and Y values for the transformation point are displayed in the Property inspector for the symbol.



Info panel with center square in coordinate grid selected; x and y coordinates of selection center point displayed

By default, the upper left square in the coordinate grid in the Info panel is selected, and the X and Y values display the location of the upper left corner of the current selection, relative to the upper left corner of the Stage.

Note: For symbol instances, the coordinate grid and the X and Y values display the location of the symbol registration point, or the location of the upper left corner of the symbol instance. See "Editing symbols" on page 61.

Transforming objects freely

You can use the Free Transform tool to freely transform objects, groups, instances, or text blocks. You can perform individual transformations or combine several transformations, such as moving, rotating, scaling, skewing, and distortion.

To transform freely:

1. Select a graphic object, instance, group, or text block on the Stage.
2. Click the Free Transform tool.

   Moving the pointer over and around the selection changes the pointer to indicate which transformation function is available.
3. Drag the handles to transform the selection, as follows:
   - To move the selection, position the pointer over the object within the bounding box, and drag the object to a new position. Do not drag the transformation point.
   - To set the center of rotation or scaling, drag the transformation point to a new location.
   - To rotate the selection, position the pointer just outside a corner handle and drag. The selection rotates around the transformation point.

     Shift-drag to rotate in 45° increments.

     Alt-drag (Windows) or Option-drag (Macintosh) to rotate around the opposite corner.
   - To scale the selection, drag a corner handle diagonally to scale in two dimensions. Drag a corner handle or a side handle horizontally or vertically to scale in the respective direction only. Shift-drag to resize proportionally.

- To skew the selection, position the pointer on the outline between the transformation handles and drag.
- To distort shapes, press Control (Windows) or Command (Macintosh) and drag a corner handle or a side handle. Shift-Control-drag (Windows) or Shift-Command-drag (Macintosh) a corner handle to *taper* the object—to move the selected corner and the adjoining corner equal distances from their origins. For more information on distorting objects, see "Distorting objects" on page 152.

Note: The Free Transform tool cannot transform symbols, bitmaps, video objects, sounds, gradients, or text. If a multiple selection contains any of these, only the shape objects are distorted. To transform text, first convert the characters to shape objects.

4. To end the transformation, click outside the selected object, instance, or text block.

Distorting objects

When you apply a Distort transformation to a selected object, dragging a corner handle or an edge handle on the bounding box moves the corner or edge and realigns the adjoining edges. Shift-dragging a corner point *tapers* the object—that is, it moves that corner and the adjoining corner an equal distance and opposite direction from each other. The adjoining corner is the corner opposite the direction you drag. Control-dragging (Windows) or Command-dragging a middle point on an edge moves the entire edge freely.

You can distort graphic objects by using the Distort command. You can also distort objects when freely transforming them. See "Transforming objects freely" on page 151.

To distort graphic objects:

1. Select a graphic object or objects on the Stage.

   Note: The Distort command cannot modify symbols, bitmaps, video objects, sounds, gradients, object groups, or text. If a multiple selection contains any of these, only the shape objects are distorted. To modify text, first convert the characters to shape objects.

2. Select Modify > Transform > Distort.
3. Place the pointer on one of the transformation handles and drag.
4. To end the transformation, click outside the selected object or objects.

Modifying shapes with the Envelope modifier

The Envelope modifier lets you warp and distort objects. An envelope is a bounding box that contains one or more objects. Changes made to an envelope's shape affect the shape of the objects contained within the envelope. You edit the shape of an envelope by adjusting its points and tangent handles. See "Adjusting segments" on page 94.

To modify a shape with the Envelope modifier:

1. Select a shape on the Stage.

   Note: The Envelope modifier cannot modify symbols, bitmaps, video objects, sounds, gradients, object groups, or text. If a multiple selection contains any of these, only the shape objects are distorted. To modify text, first convert the characters to shape objects.

2. Select Modify > Transform > Envelope.
3. Drag the points and tangent handles to modify the envelope.

Scaling objects

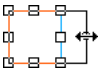
Scaling an object enlarges or reduces the object horizontally, vertically, or both. You can scale an object by dragging or by entering values in the Transform panel.

To scale objects by dragging:

1. Select a graphic object or objects on the Stage.
2. Select Modify > Transform > Scale.
3. Do one of the following:
   - To scale the object both horizontally and vertically, drag one of the corner handles. Proportions are maintained as you scale. Shift-drag to scale nonuniformly.

   - To scale the object either horizontally or vertically, drag a center handle.

4. To end the transformation, click outside the selected object or objects.

   Note: When you increase the size of a number of items, those near the edges of the bounding box might be moved off the Stage. If this occurs, select View › Work Area to see the elements that are beyond the edges of the Stage.

Rotating objects

Rotating an object turns it around its transformation point. The transformation point is aligned with the registration point, which defaults to the center of the object, but you can move the point by dragging it. You can rotate an object by using the Rotate commands, by dragging with the Free Transform tool, or by specifying an angle in the Transform panel. When you rotate an object by dragging, you can also skew and scale the object in the same operation. When you rotate an object using the Transform panel, you can scale the object in the same operation.

To rotate and skew objects by dragging:

1. Select the object or objects on the Stage.
2. Select Modify > Transform > Rotate and Skew.
3. Do one of the following:
   - Drag a corner handle to rotate the object.
   - Drag a center handle to skew the object.
4. To end the transformation, click outside the selected object or objects.

To rotate objects by 90°:

1. Select the object or objects.
2. Select Modify > Transform > Rotate 90° CW to rotate clockwise, or Rotate 90° CCW to rotate counterclockwise.

Skewing objects

Skewing an object transforms it by slanting it along one or both axes. You can skew an object by dragging or by entering a value in the Transform panel. To skew an object by dragging, see the procedure for rotating and skewing an object by dragging, under "Rotating objects" on page 153.

To skew an object using the Transform panel:

1. Select the object or objects.
2. Select Window > Design Panels > Transform.
3. Click Skew.
4. Enter angles for the horizontal and vertical values.

Flipping objects

You can flip objects across their vertical or horizontal axis without moving their relative position on the Stage.

To flip an object:

1. Select the object.
2. Select Modify > Transform > Flip Vertical or Flip Horizontal.

Restoring transformed objects

When you scale, rotate, and skew instances, groups, and type with the Transform panel, Flash saves the original size and rotation values with the object. This allows you to remove the transformations you applied and restore the original values.

You can undo only the most recent transformation performed in the Transform panel by choosing Edit > Undo. You can reset all transformations performed in the Transform panel by clicking the Reset button in the panel before you deselect the object.

To restore a transformed object to its original state:

1. Select the transformed object.
2. Select Modify > Transform > Remove Transform.

To reset a transformation performed in the Transform panel:

- With the transformed object still selected, click the Reset button in the Transform panel.

Aligning objects

The Align panel enables you to align selected objects along the horizontal or vertical axis. You can align objects vertically along the right edge, center, or left edge of the selected objects, or horizontally along the top edge, center, or bottom edge of the selected objects. Edges are determined by the bounding boxes enclosing each selected object.

Using the Align panel, you can distribute selected objects so that their centers or edges are evenly spaced. You can resize selected objects so that the horizontal or vertical dimensions of all objects match those of the largest selected object. You can also align selected objects to the Stage. You can apply one or more Align options to selected objects.

To align objects:

1. Select the objects to align.
2. Select Window > Design Panels > Align.
3. In the Align panel, select To Stage to apply alignment modifications relative to stage dimensions.
4. Select alignment buttons to modify the selected objects:
   - For Align, select Align Left, Align Horizontal Center, Align Right, Align Top, Align Vertical Center, or Align Bottom.
   - For Distribute, select Distribute Top, Distribute Horizontal Center, Distribute Bottom, Distribute Left, Distribute Vertical Center, or Distribute Right.
   - For Match Size, select Match Width, Match Height, or Match Width and Height.
   - For Space, select Space Horizontally or Space Vertically.

Breaking apart groups and objects

To separate groups, instances, and bitmaps into ungrouped, editable elements, you use the Break Apart command. Breaking apart significantly reduces the file size of imported graphics.

Although you can select Edit > Undo immediately after breaking apart a group or object, breaking apart is not entirely reversible. It affects objects as follows:

- It severs a symbol instance's link to its master symbol.
- It discards all but the current frame in an animated symbol.
- It converts a bitmap to a fill.

- It places each character into a separate text block when applied to text blocks.
- It converts characters to outlines when applied to a single text character. See "Breaking text apart" on page 120.

The Break Apart command should not be confused with the Ungroup command. The Ungroup command separates grouped objects, returning grouped elements to the state they were in prior to grouping. It does not break apart bitmaps, instances, or type, or convert type to outlines.

To break apart groups or objects:

1. Select the group, bitmap, or symbol that you want to break apart.
2. Select Modify > Break Apart.

   Note: Breaking apart animated symbols, or groups within an interpolated animation, is not recommended and might have unpredictable results. Breaking apart complex symbols and large blocks of text can take a long time. You might need to increase the application's memory allocation to properly break apart complex objects.

CHAPTER 9
Creating Motion

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 offer several ways to include animation and special effects in your document. Timeline effects, such as blur, expand, and explode, make it easy to animate an object: you can simply select the object, then select an effect and specify parameters. With Timeline effects, you can accomplish in a few easy steps a previously time-consuming task that required more advanced knowledge of animation.

To create tweened animation, you create starting and ending frames and let Flash create the animation for the frames in between. Flash varies the object's size, rotation, color, or other attributes between the starting and ending frames to create the appearance of movement. See "Tweened animation" on page 161.

You can also create animation by changing the contents of successive frames in the Timeline. You can make an object move across the Stage, increase or decrease its size, rotate, change color, fade in or out, or change shape. Changes can occur independently of, or in concert with, other changes. For example, you can make an object rotate and fade in as it moves across the Stage. In frame-by-frame animation, you create the image in every frame. See "Frame-by-frame animation" on page 162.

This chapter contains the following sections:

Using Timeline effects

Flash includes prebuilt Timeline effects that allow you to create complex animations with a minimal number of steps. You can apply Timeline effects to the following objects:

- Text
- Graphics, including shapes, groups, and graphic symbols
- Bitmap images
- Button symbols

Note: When you apply a Timeline effect to a movie clip, Flash nests the effect within the movie clip.

Adding a Timeline effect

When you add a Timeline effect to an object, Flash creates a layer and transfers the object to the new layer. The object is placed inside the effect graphic, and all tweens and transformations required for the effect reside in the graphic on the newly created layer.

The new layer automatically receives the same name as the effect, appended with a number that represents the order in which the effect is applied, out of all effects in your document.

When you add a Timeline effect, a folder with the effect's name is added to the library, containing elements used in creating the effect.

To add an effect to an object:

1. Do one of the following to add a Timeline effect:
   - Select the object to which you're adding the Timeline effect. Select Insert > Timeline Effects. Then select Assistants, Effects, or Transition/Transform from the submenu, and select an effect from the list.
   - Right-click (Windows) or Control-click (Macintosh) the object to which you're adding the Timeline effect. From the context menu, select Timeline Effects. Then select Assistants, Effects, or Transition/Transform from the submenu, and select an effect from the list.

   Effects available for the type of object you've selected appear as active menu choices.
2. In the dialog box that appears for the effect, view the effect preview based on default settings. Modify the default settings as desired, and then click Update Preview to view the effect with the new settings. For more information, see the next section.
3. When the Timeline effect appears as desired in the preview window, click OK.

Timeline effect settings

Each Timeline effect manipulates a graphic or symbol in a specific way and allows you to change individual parameters for a desired effect. In the preview window, you can quickly see the changes made when you alter settings.

Motion effect name and description	Settings
Copy to grid	
Duplicates a selected object by the number of columns and then multiplies the columns by the number of rows to create a grid of the elements.	• Number of rows • Number of columns • Distance between rows, in pixels • Distance between columns, in pixels
Distributed duplication	
Duplicates a selected object the number of times entered in the settings. The first element is a copy of the original object. The objects are modified in increments until the final object reflects the parameters entered in the settings.	• Number of copies • Offset distance, x position, in pixels • Offset distance, y position, in pixels • Offset rotation, in degrees • Offset start frame, in frames across Timeline • Exponential scaling by x, y scale, in delta percentage • Linear scaling by x, y scale, in delta percentage • Final alpha, in percentage • Change color, select/deselect • Final color, in RGB hex value (final copy has this color value; intermediate copies gradually transition to it) • Duplication delay, in frames (results in pause between copies)
Blur	
Creates a motion blur effect by changing the alpha value, position, or scale of an object over time.	• Effect duration, in frames • Allow horizontal blur • Allow vertical blur • Direction of blur • Number of steps • Starting scale
Drop shadow	
Creates a shadow below the selected element.	• Color, in hex RGB value • Alpha transparency, in percentage • Shadow offset, in *x, y* offset, in pixels

Motion effect name and description	Settings
Expand	
Expands, contracts, or expands and contracts objects over time. This effect works best with two or more objects grouped together or combined in a movie clip or graphic symbol. Objects containing text or letters work well with this effect.	• Expand duration, in frames • Expand, squeeze, both • Expand direction, to left, from center, to right • Fragment offset, in pixels • Shift group center by, *x, y* offset, in pixels • Change fragment size by, height, width, in pixels
Explode	
Gives the illusion of an object exploding. Elements of text or a complex group of objects (symbols, shapes or video clips) break apart, spin, and arc outward.	• Effect duration, in frames • Direction of explosion, upward to left, center, or right, downward to left, center, or right • Arc size, *x, y* offset in pixels • Rotate fragments by, in degrees • Change fragments size by, in degrees • Final alpha, in percentage
Transform	
Adjusts the position, scale, rotation, alpha, and tint of the selected elements. Use Transform to apply a single effect or a combination of effects to create Fade In/Out, Fly In/Out, Grow/Shrink, and Spin Left/Right effects.	• Effect duration, in frames • Move to position, *x, y* offset, in pixels • Change position by, *x, y* offset, in pixels • Scale, lock to equally apply change, in percentage, unlock to apply *x* and/or *y* axis change separately, in percentage • Rotate, in degrees • Spin, number of times • Times, counterclockwise, clockwise • Change color, select/deselect • Final color, in RGB hex value • Final alpha, in percentage • Motion ease
Transition	
Wipes in or wipes out selected objects by fading, wiping, or a combination of both.	• Effect duration, in frames • Direction, toggle between in (coming in) and out (going out), select up, down, left, or right • Fade, select/deselect • Wipe, select/deselect • Motion ease

Editing a Timeline effect

You can edit Timeline effects using the Effect Settings dialog box.

To edit a Timeline effect:

1. Select the object associated with the effect on the Stage.
2. To open the Effect Settings dialog box, do one of the following:
   - In the Property inspector, click Edit.
   - Right-click (Windows) or Control-click (Macintosh) the object and select Timeline Effects > Edit Effect from the context menu.
3. In the Effect Settings dialog box, edit the settings as desired, and then click OK.

Deleting a Timeline effect

You use the context menu to delete Timeline effects.

To delete a Timeline effect:

- On the Stage, right-click (Windows) or Control-click (Macintosh) the object that has the Timeline effect that you want to remove, and select Timeline Effects > Remove Effect from the context menu.

Tweened animation

Flash can create two types of tweened animation, *motion tweening* and *shape tweening*.

- In motion tweening, you define properties such as position, size, and rotation for an instance, group, or text block at one point in time, and then you change those properties at another point in time. You can also apply a motion tween along a path. See "Tweening instances, groups, and type" on page 165 and "Tweening motion along a path" on page 168.
- In shape tweening, you draw a shape at one point in time, and then you change that shape or draw another shape at another point in time. Flash interpolates the values or shapes for the frames in between, creating the animation. See "Tweening shapes" on page 169.

Note: To apply shape tweening to groups, instances, or bitmap images, you must first break these elements apart. See "Breaking apart groups and objects" on page 155. To apply shape tweening to text, you must break the text apart twice to convert the text to objects. See "Breaking text apart" on page 120.

Tweened animation is an effective way to create movement and changes over time while minimizing file size. In tweened animation, Flash stores only the values for the changes between frames.

To quickly prepare elements in a document for tweened animation, distribute objects to layers. See "Distributing objects to layers" on page 165.

You can apply tweened animation to an object on a mask layer to create a dynamic mask. For information on mask layers, see "Using mask layers" on page 174.

Frame-by-frame animation

Frame-by-frame animation changes the contents of the Stage in every frame and is best suited to complex animation in which an image changes in every frame instead of simply moving across the Stage. Frame-by-frame animation increases file size more rapidly than tweened animation. In frame-by-frame animation, Flash stores the values for each complete frame. For information on frame-by-frame animations, see "Creating frame-by-frame animations" on page 171.

Layers in animation

Each scene in a Flash document can consist of any number of layers. As you animate, you use layers and layer folders to organize the components of an animation sequence and to separate animated objects so they don't erase, connect, or segment each other. If you want Flash to tween the movement of more than one group or symbol at once, each must be on a separate layer. Typically, the background layer contains static artwork, and each additional layer contains one separate animated object.

When a document has several layers, tracking and editing the objects on one or two of them can be difficult. This task is easier if you work with the contents of one layer at a time. Layer folders help you organize layers into manageable groups that you can expand and collapse to view only the layers relevant to your current task. See "Using layers" in *Getting Started with Flash*.

Creating keyframes

A keyframe is a frame where you define changes in the animation. When you create frame-by-frame animation, every frame is a keyframe. In tweened animation, you define keyframes at significant points in the animation and let Flash create the contents of frames in between. Flash displays the interpolated frames of a tweened animation as light blue or light green with an arrow drawn between keyframes. Because Flash documents save the shapes in each keyframe, you should create keyframes only at those points in the artwork where something changes.

Keyframes are indicated in the Timeline: a keyframe with content on it is represented by a solid circle, and an empty keyframe is represented by an empty circle before the frame. Subsequent frames that you add to the same layer have the same content as the keyframe.

To create a keyframe, do one of the following:

- Select a frame in the Timeline and select Insert > Timeline > Keyframe.
- Right-click (Windows) or Control-click (Macintosh) a frame in the Timeline and select Insert Keyframe.

Representations of animations in the Timeline

Flash distinguishes tweened animation from frame-by-frame animation in the Timeline as follows:

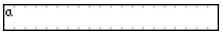
- Motion tweens are indicated by a black dot at the beginning keyframe; intermediate tweened frames have a black arrow with a light blue background.

- Shape tweens are indicated by a black dot at the beginning keyframe; intermediate frames have a black arrow with a light green background.

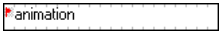
- A dashed line indicates that the tween is broken or incomplete, such as when the final keyframe is missing.

- A single keyframe is indicated by a black dot. Light gray frames after a single keyframe contain the same content with no changes and have a black line with a hollow rectangle at the last frame of the span.

- A small *a* indicates that the frame has been assigned a frame action with the Actions panel.



- A red flag indicates that the frame contains a label or comment.



- A gold anchor indicates that the frame is a named anchor.

Frame rates

The frame rate, the speed at which the animation is played, is measured in number of frames per second. A frame rate that's too slow makes the animation appear to stop and start; a frame rate that's too fast blurs the details of the animation. A frame rate of 12 frames per second (fps) usually gives the best results on the web. QuickTime and AVI movies generally have a frame rate of 12 fps, while the standard motion-picture rate is 24 fps.

The complexity of the animation and the speed of the computer on which the animation is being played affect the smoothness of the playback. Test your animations on a variety of machines to determine optimum frame rates.

Because you specify only one frame rate for the entire Flash document, it's a good idea to set this rate before you begin creating animation. See "Creating or opening a document and setting properties" on page 12.

Extending still images

When you create a background for animation, it's often necessary that a still image remain the same for several frames. Adding a span of new frames (not keyframes) to a layer extends the contents of the last keyframe in all the new frames.

To extend a still image through multiple frames:

1. Create an image in the first keyframe of the sequence.
2. Select a frame to the right, marking the end of the span of frames that you want to add.
3. Select Insert > Timeline > Frame.

To use a shortcut to extend still images:

1. Create an image in the first keyframe.
2. Alt-drag (Windows) or Option-drag (Macintosh) the keyframe to the right. This creates a span of new frames, but without a new keyframe at the end point.

Distributing objects to layers for tweened animation

You can quickly distribute selected objects in a frame to separate layers to apply tweened animation to the objects. The objects can be on one or more layers initially. Flash distributes each object to a new, separate layer. Any objects that you don't select (including objects in other frames) are preserved in their original positions.

You can apply the Distribute to Layers command to any type of element on the Stage, including graphic objects, instances, bitmaps, video clips, and broken-apart text blocks.

Applying the Distribute to Layers command to broken-apart text makes it easy to create animated text. The characters in the text are placed in separate text blocks during the Break Apart operation, and each text block is placed on a separate layer during the Distribute to Layers process. For information on breaking text apart, see "Breaking text apart" on page 120.

New layers

New layers created during the Distribute to Layers operation are named according to the name of the element that each contains:

- A new layer containing a library asset (such as a symbol, bitmap, or video clip) is given the same name as the asset.
- A new layer containing a named instance is given the name of the instance.
- A new layer containing a character from a broken-apart text block is named with the character.
- A new layer containing a graphic object (which has no name) is named Layer1 (or Layer2, and so on), because graphic objects do not have names.

Flash inserts new layers below any selected layers in the Timeline. The new layers are arranged top to bottom, in the order in which the selected elements were originally created. For broken-apart text, the layers are arranged in the order of the characters, whether left-to-right, right-to-left, or top-to-bottom. For example, if you break apart the text *FLASH* and distribute it to layers, the new layers, named F, L, A, S, and H, are arranged top to bottom, immediately below the layer initially containing the text.

Distributing objects to layers

To distribute objects to layers, you select the objects in one or more layers and select Distribute to Layers from the Modify menu or from the context menu.

To tween distributed objects, follow the procedure in "Tweening instances, groups, and type" on page 165 or "Tweening shapes" on page 169.

To distribute objects to layers:

1. Select the objects that you want to distribute to layers. The objects can be in a single layer, or in several layers, including noncontiguous layers.
2. Do one of the following:
   - Select Modify > Timeline > Distribute to Layers.
   - Right-click (Windows) or Control-click (Macintosh) one of the selected objects and select Distribute to Layers from the context menu.

Tweening instances, groups, and type

To tween the changes in properties of instances, groups, and type, you use motion tweening. Flash can tween position, size, rotation, and skew of instances, groups, and type. Additionally, Flash can tween the color of instances and type, creating gradual color shifts or making an instance fade in or out. To tween the color of groups or type, you must make them into symbols. See "Creating symbols" on page 55. To animate individual characters in a block of text separately, you place each character in a separate text block; see "Breaking text apart" on page 120.

If you apply a motion tween and then change the number of frames between the two keyframes, or move the group or symbol in either keyframe, Flash automatically tweens the frames again.

You can create a motion tween using one of two methods:

- Create the starting and ending keyframes for the animation and use the Motion Tweening option in the Property inspector.
- Create the first keyframe for the animation, insert the number of frames you want on the Timeline, select Insert > Timeline > Create Motion Tween, and move the object to the new location on the Stage. Flash automatically creates the ending keyframe.

When tweening position, you can make the object move along a nonlinear path. See "Tweening motion along a path" on page 168.

To create a motion tween using the Motion Tweening option:

1. Click a layer name to make it the active layer, and select an empty keyframe in the layer where you want the animation to start.
2. To create the first frame of the motion tween, do one of the following:
   - Create a graphic object with the Pen, Oval, Rectangle, Pencil, or Brush tool, then convert it to a symbol. For more information on converting objects to symbols, see "Creating symbols" on page 55.
   - Create an instance, group, or text block on the Stage.
   - Drag an instance of a symbol from the Library panel.
3. Create a second keyframe where you want the animation to end, then select the ending frame (immediately to the left of the second keyframe on the Timeline).
4. Do any of the following to modify the instance, group, or text block in the ending frame:
   - Move the item to a new position.
   - Modify the item's size, rotation, or skew.
   - Modify the item's color (instance or text block only).

   To tween the color of elements other than instances or text blocks, use shape tweening. See "Tweening shapes" on page 169.
5. Click any frame in the tween's frame span and select Motion from the Tween pop-up menu in the Property inspector (Window > Properties).
6. If you modified the size of the item in step 4, select Scale to tween the size of the selected item.
7. Drag the arrow next to the Easing value or enter a value to adjust the rate of change between tweened frames:
   - To begin the motion tween slowly and accelerate the tween toward the end of the animation, drag the slider up or enter a negative value between -1 and -100.
   - To begin the motion tween rapidly and decelerate the tween toward the end of the animation, drag the slider down or enter a positive value between 1 and 100.

   By default, the rate of change between tweened frames is constant. Easing creates a more natural appearance of acceleration or deceleration by gradually adjusting the rate of change.
8. To rotate the selected item while tweening, select an option from the Rotate menu:
   - Select None (the default setting) to prevent rotation.
   - Select Auto to rotate the object once in the direction requiring the least motion.
   - Select Clockwise (CW) or Counterclockwise (CCW) to rotate the object as indicated, and then enter a number to specify the number of rotations.

   Note: The rotation in step 9 is in addition to any rotation you applied to the ending frame in step 4.
9. If you're using a motion path, select Orient to Path to orient the baseline of the tweened element to the motion path. (See "Tweening motion along a path" on page 168.)

10. Select the Sync option in the Property inspector to synchronize the animation of graphic symbol instances with the main Timeline.

    Note: Modify › Timeline › Synchronize Symbols and the Sync option both recalculate the number of frames in a tween to match the number of frames allotted to it in the Timeline.

11. If you're using a motion path, select Snap to attach the tweened element to the motion path by its registration point.

To create a motion tween using the Create Motion Tween command:

1. Select an empty keyframe and draw an object on the Stage, or drag an instance of a symbol from the Library panel.

   Note: To create a tween, you must have only one item on the layer.

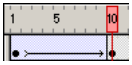
2. Select Insert > Timeline > Create Motion Tween.

   If you drew an object in step 1, Flash automatically converts the object to a symbol and assigns it the name tween1.

3. Click inside the frame where you want the animation to end, and select Insert > Timeline > Frame.

4. Move the object, instance, or text block on the Stage to the desired position. Adjust the size of the element if you want to tween its scale. Adjust the rotation of the element if you want to tween its rotation. Deselect the object when you have completed adjustments.

   A keyframe is automatically added to the end of the frame range.



5. Drag the arrow next to the Easing value or enter a value to adjust the rate of change between tweened frames:

   - To begin the motion tween slowly and accelerate the tween toward the end of the animation, drag the slider up or enter a value between -1 and -100.
   - To begin the motion tween rapidly and decelerate the tween toward the end of the animation, drag the slider down or enter a positive value between 1 and 100.

   By default, the rate of change between tweened frames is constant. Easing creates a more natural appearance of acceleration or deceleration by gradually adjusting the rate of change.

6. To rotate the selected item while tweening, select an option from the Rotate menu:

   - Select Auto to rotate the object once in the direction requiring the least motion.
   - Select Clockwise (CW) or Counterclockwise (CCW) to rotate the object as indicated, and then enter a number to specify the number of rotations.

   Note: The rotation in step 6 is in addition to any rotation you applied to the ending frame in step 4.

7. If you're using a motion path, select Orient to Path to orient the baseline of the tweened element to the motion path. (See "Tweening motion along a path" on page 168.)

8. Select Synchronize to ensure that the instance loops properly in the main document.

   Use the Synchronize command if the number of frames in the animation sequence inside the symbol is not an even multiple of the number of frames the graphic instance occupies in the document.

9. If you're using a motion path, select Snap to attach the tweened element to the motion path by its registration point.

Tweening motion along a path

Motion guide layers let you draw paths along which tweened instances, groups, or text blocks can be animated. You can link multiple layers to a motion guide layer to have multiple objects follow the same path. A normal layer that is linked to a motion guide layer becomes a guided layer.

To create a motion path for a tweened animation:

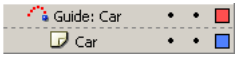
1. Create a motion-tweened animation sequence as described in "Tweening instances, groups, and type" on page 165.

   If you select Orient to Path, the baseline of the tweened element orients to the motion path. If you select Snap, the registration point of the tweened element snaps to the motion path.

2. Do one of the following:

   - Select the layer containing the animation and select Insert > Timeline > Motion Guide.
   - Right-click (Windows) or Control-click (Macintosh) the layer containing the animation and select Add Motion Guide from the context menu.

   Flash creates a new layer above the selected layer with a motion guide icon to the left of the layer name.



3. Use the Pen, Pencil, Line, Circle, Rectangle, or Brush tool to draw the desired path.

4. Snap the center to the beginning of the line in the first frame, and to the end of the line in the last frame.

   Note: For best snapping results, drag the symbol by its registration point.

5. To hide the motion guide layer and the line so that only the object's movement is visible while you work, click in the Eye column on the motion guide layer.

   The group or symbol follows the motion path when you play the animation.

To link layers to a motion guide layer, do one of the following:

- Drag an existing layer below the motion guide layer. The layer is indented under the motion guide layer. All objects on this layer automatically snap to the motion path.
- Create a new layer under the motion guide layer. Objects you tween on this layer are automatically tweened along the motion path.
- Select a layer below a motion guide layer. Select Modify > Timeline > Layer Properties and select Guided in the Layer Properties dialog box.

To unlink layers from a motion guide layer:

1. Select the layer you want to unlink.
2. Do one of the following:
   - Drag the layer above the motion guide layer.
   - Select Modify > Timeline > Layer Properties and select Normal as the layer type in the Layer Properties dialog box.

Tweening shapes

By tweening shapes, you can create an effect similar to morphing, making one shape appear to change into another shape over time. Flash can also tween the location, size, color, and opacity of shapes.

Tweening one shape at a time usually yields the best results. If you tween multiple shapes at one time, all the shapes must be on the same layer.

To apply shape tweening to groups, instances, or bitmap images, you must first break these elements apart. See "Breaking apart groups and objects" on page 155. To apply shape tweening to text, you must break the text apart twice to convert the text to objects. See "Breaking text apart" on page 120.

To control more complex or improbable shape changes, you use shape hints, which control how parts of the original shape move into the new shape. See "Using shape hints" on page 170.

To tween a shape:

1. Click a layer name to make it the active layer, and create or select a keyframe where you want the animation to start.
2. Create or place the artwork for the first frame of the sequence. For best results, the frame should contain only one item (a graphic object or broken-apart group, bitmap, instance, or text block).
3. Select the keyframe in the Timeline.
4. Select Window > Properties.
5. In the Property inspector, select Shape from the Tween pop-up menu.

6. Drag the arrow next to the Easing value or enter a value to adjust the rate of change between tweened frames:
   - To begin the shape tween gradually and accelerate the tween toward the end of the animation, drag the slider down or enter a negative value between -1 and -100.
   - To begin the shape tween rapidly and decelerate the tween toward the end of the animation, drag the slider up or enter a positive value between 1 and 100.

   By default, the rate of change between tweened frames is constant. Easing creates a more natural appearance of transformation by gradually adjusting the rate of change.
7. Select an option for Blend:

   Distributive creates an animation in which the intermediate shapes are smoother and more irregular.

   Angular creates an animation that preserves apparent corners and straight lines in the intermediate shapes.

   Note: Angular is appropriate only for blending shapes with sharp corners and straight lines. If the shapes you select do not have corners, Flash reverts to distributive shape tweening.
8. Create a second keyframe the desired number of frames after the first keyframe.
9. With the second keyframe selected, select the artwork you placed in the first keyframe and do one of the following:
   - Modify the shape, color, opacity, or position of the artwork.
   - Delete the artwork and place new artwork in the second keyframe.

Using shape hints

To control more complex or improbable shape changes, you can use shape hints. Shape hints identify points that should correspond in starting and ending shapes. For example, if you are tweening a drawing of a face as it changes expression, you can use a shape hint to mark each eye. Then, instead of the face becoming an amorphous tangle while the shape change takes place, each eye remains recognizable and changes separately during the shift.

Shape hints contain letters (*a* through *z*) for identifying which points correspond in the starting and ending shape. You can use up to 26 shape hints.

Shape hints are yellow in a starting keyframe, green in an ending keyframe, and red when not on a curve.

For best results when tweening shapes, follow these guidelines:

- In complex shape tweening, create intermediate shapes and tween them instead of just defining a starting and ending shape.
- Make sure that shape hints are logical. For example, if you're using three shape hints for a triangle, they must be in the same order on the original triangle and on the triangle to be tweened. The order cannot be *abc* in the first keyframe and *acb* in the second.
- Shape hints work best if you place them in counterclockwise order beginning at the top left corner of the shape.

To use shape hints:

1. Select the first keyframe in a shape-tweened sequence.
2. Select Modify > Shape > Add Shape Hint.

   The beginning shape hint appears as a red circle with the letter *a* somewhere on the shape.
3. Move the shape hint to a point that you want to mark.
4. Select the last keyframe in the tweening sequence.

   The ending shape hint appears somewhere on the shape as a green circle with the letter *a*.
5. Move the shape hint to the point in the ending shape that should correspond to the first point you marked.
6. Play the animation again to see how the shape hints change the shape tweening. Move the shape hints to fine-tune the tweening.
7. Repeat this process to add additional shape hints. New hints appear with the letters that follow (*b, c,* and so on).

You can choose to view all shape hints, and you can remove shape hints.

To see all shape hints:

- Select View > Show Shape Hints. The layer and keyframe that contain shape hints must be active for Show Shape Hints to be available.

To remove a shape hint:

- Drag it off the Stage.

To remove all shape hints:

- Select Modify > Shape > Remove All Hints.

Creating frame-by-frame animations

To create a frame-by-frame animation, you define each frame as a keyframe and create a different image for each frame. Each new keyframe initially contains the same contents as the keyframe preceding it, so you can modify the frames in the animation incrementally.

To create a frame-by-frame animation:

1. Click a layer name to make it the active layer, and select a frame in the layer where you want the animation to start.
2. If the frame isn't already a keyframe, select Insert > Timeline > Keyframe to make it one.
3. Create the artwork for the first frame of the sequence.

   You can use the drawing tools, paste graphics from the Clipboard, or import a file.
4. Click the next frame to the right in the same row and select Insert > Timeline > Keyframe, or right-click (Windows) or Control-click (Macintosh) and select Insert Keyframe from the context menu.

   This adds a new keyframe whose contents are the same as those of the first keyframe.
5. Alter the contents of this frame on the Stage to develop the next increment of the animation.

6. To complete your frame-by-frame animation sequence, repeat steps 4 and 5 until you've built the motion you want.
7. To test the animation sequence, select Control > Play or click the Play button on the Controller.

Editing animation

After you create a frame or a keyframe, you can move it elsewhere in the active layer or to another layer, remove it, and make other changes. Only keyframes are editable. You can view tweened frames, but you can't edit them directly. To edit tweened frames, you change one of the defining keyframes or insert a new keyframe between the beginning and ending keyframes. You can drag items from the Library panel onto the Stage to add the items to the current keyframe.

To display and edit more than one frame at a time, you use onion skinning. See "Onion skinning" on page 173.

To insert frames in the Timeline, do one of the following:

- To insert a new frame, select Insert > Timeline > Frame.
- To create a new keyframe, select Insert > Timeline > Keyframe, or right-click (Windows) or Control-click (Macintosh) the frame where you want to place a keyframe, and select Insert Keyframe from the context menu.
- To create a new blank keyframe, select Insert > Timeline > Blank Keyframe, or right-click (Windows) or Control-click (Macintosh) the frame where you want to place the keyframe, and select Insert Blank Keyframe from the context menu.

To delete or modify a frame or keyframe, do one of the following:

- To delete a frame, keyframe, or frame sequence, select the frame, keyframe, or sequence and right-click (Windows) or Control-click (Macintosh) the frame, keyframe, or sequence and select Remove Frames from the context menu. Surrounding frames remain unchanged.
- To move a keyframe or frame sequence and its contents, select the keyframe or sequence, then drag to the desired location.
- To extend the duration of a keyframe, Alt-drag (Windows) or Option-drag (Macintosh) the keyframe to the final frame of the new sequence.
- To copy a keyframe or frame sequence by dragging, select the keyframe or sequence, then Alt-drag (Windows) or Option-drag (Macintosh) to the new location.
- To copy and paste a frame or frame sequence, select the frame or sequence and select Edit > Timeline > Copy Frames. Select a frame or sequence that you want to replace, and select Edit > Timeline > Paste Frames.
- To convert a keyframe to a frame, select the keyframe and select Modify > Timeline > Clear Keyframe, or right-click (Windows) or Control-click (Macintosh) the keyframe and select Clear Keyframe from the context menu. The cleared keyframe and all frames up to the subsequent keyframe are replaced with the contents of the frame preceding the cleared keyframe.

- To change the length of a tweened sequence, drag the beginning or ending keyframe left or right. To change the length of a frame-by-frame sequence, see "Creating frame-by-frame animations" on page 171.
- To add a library item to the current keyframe, drag the item from the Library panel onto the Stage.
- To reverse an animation sequence, select the appropriate frames in one or more layers and select Modify > Timeline > Reverse Frames. There must be keyframes at the beginning and end of the sequence.

Onion skinning

Normally, Flash displays one frame of the animation sequence at a time on the Stage. To help you position and edit a frame-by-frame animation, you can view two or more frames on the Stage at once. The frame under the playhead appears in full color, while surrounding frames are dimmed, making it appear as if each frame were drawn on a sheet of translucent onion-skin paper and the sheets were stacked on top of each other. Dimmed frames cannot be edited.

To simultaneously see several frames of an animation on the Stage:

- Click the Onion Skin button. All frames between the Start Onion Skin and End Onion Skin markers (in the Timeline header) are superimposed as one frame in the Document window.

To control onion skinning display, do any of the following:

- To display onion skinned frames as outlines, click the Onion Skin Outlines button.
- To change the position of either onion skin marker, drag its pointer to a new location. (Normally, the onion skin markers move in conjunction with the current frame pointer.)
- To enable editing of all frames between onion skin markers, click the Edit Multiple Frames button. Usually onion skinning lets you edit only the current frame. However, you can display the contents of each frame between the onion skin markers normally, and make each available for editing, regardless of which is the current frame.

  Note: Locked layers (those with a padlock icon) aren't displayed when onion skinning is turned on. To avoid a multitude of confusing images, you can lock or hide the layers you don't want onion skinned.

To change the display of onion skin markers:

- Click the Modify Onion Markers button and select an item from the menu:

  Always Show Markers displays the onion skin markers in the Timeline header whether or not onion skinning is on.

  Anchor Onion locks the onion skin markers to their current position in the Timeline header. Normally, the Onion Skin range is relative to the current frame pointer and the Onion Skin markers. By anchoring the Onion Skin markers, you prevent them from moving with the current frame pointer.

  Onion 2 displays two frames on either side of the current frame.

  Onion 5 displays five frames on either side of the current frame.

  Onion All displays all frames on either side of the current frame.

Moving an entire animation

If you need to move an entire animation on the Stage, you must move the graphics in all frames and layers at once to avoid realigning everything.

To move the entire animation to another location on the Stage:

1. Unlock all layers.

   To move everything on one or more layers but nothing on other layers, lock or hide all the layers you don't want to move.
2. Click the Edit Multiple Frames button in the Timeline.
3. Drag the onion skin markers so that they enclose all the frames you want to select, or click Modify Onion Markers and select Onion All.
4. Select Edit > Select All.
5. Drag the entire animation to the new location on the Stage.

Using mask layers

For spotlight effects and transitions, you can use a mask layer to create a hole through which underlying layers are visible. A mask item can be a filled shape, a type object, an instance of a graphic symbol, or a movie clip. You can group multiple layers together under a single mask layer to create sophisticated effects.

To create dynamic effects, you can animate a mask layer. For a filled shape used as a mask, you use shape tweening; for a type object, graphic instance, or movie clip, you use motion tweening. When using a movie clip instance as a mask, you can animate the mask along a motion path.

To create a mask layer, you place a mask item on the layer that you want to use as a mask. Instead of having a fill or stroke, the mask item acts as a window that reveals the area of linked layers that lie beneath it. The rest of the mask layer conceals everything except what shows through the mask item. A mask layer can contain only one mask item. You cannot have a mask layer inside a button, and you cannot apply a mask to another mask.

You can also use ActionScript to create a mask layer from a movie clip. A mask layer created with ActionScript can be applied only to another movie clip. See "Using movie clips as masks" in *Using ActionScript in Flash*.

To create a mask layer:

1. Select or create a layer containing the objects to appear inside the mask.
2. With the layer selected, select Insert > Timeline > Layer to create a new layer above it.

   A mask layer always masks the layer immediately below it, so be sure to create the mask layer in the proper place.
3. Place a filled shape, text, or an instance of a symbol on the mask layer.

   Flash ignores bitmaps, gradients, transparency, colors, and line styles in a mask layer. Any filled area is completely transparent in the mask; any nonfilled area is opaque.

4. Right-click (Windows) or Control-click (Macintosh) the mask layer's name in the Timeline, and select Mask from the context menu.

   The layer is converted to a mask layer, indicated by a mask layer icon. The layer immediately below it is linked to the mask layer, and its contents show through the filled area on the mask. The masked layer name is indented, and its icon changes to a masked layer icon.

5. To display the mask effect in Flash, lock the mask layer and the masked layer.

To mask additional layers after creating a mask layer, do one of the following:

- Drag an existing layer directly below the mask layer.
- Create a new layer anywhere below the mask layer.
- Select Modify > Timeline > Layer Properties and select Masked in the Layer Properties dialog box.

To unlink layers from a mask layer:

1. Select the layer you want to unlink.
2. Do one of the following:
   - Drag the layer above the mask layer.
   - Select Modify > Timeline > Layer Properties and select Normal.

To animate a filled shape, type object, or graphic symbol instance on a mask layer:

1. Select the mask layer in the Timeline.
2. Click in the Lock column to unlock the mask layer.
3. Do one of the following:
   - If the mask object is a filled shape, apply shape tweening to the object as described in "Tweening shapes" on page 169.
   - If the mask object is a type object or graphic symbol instance, apply motion tweening to the object as described in "Tweening instances, groups, and type" on page 165.
4. When you've completed the animation operation, click in the Lock column for the mask layer to lock the layer again.

To animate a movie clip on a mask layer:

1. Select the mask layer in the Timeline.
2. Double-click the movie clip on the Stage to edit the movie clip in place and to display the movie clip's Timeline.
3. Apply motion tweening to the movie clip as described in "Tweening instances, groups, and type" on page 165. To animate the movie clip on a motion path, see "Tweening motion along a path" on page 168.
4. When you've completed the animation procedure, click the Back button in the Edit in Place window to return to document-editing mode.
5. Click in the Lock column for the mask layer to lock the layer again.

CHAPTER 10
Working with Video

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 provide several ways for you to include video in your Flash documents:

- You can import video clips into Flash as embedded files in QuickTime video (MOV), Audio Video Interleaved file (AVI), Motion Picture Experts Group file (MPEG), or other formats, depending on your system. As with an imported bitmap or vector artwork file, an embedded video file becomes part of the Flash document. For information on file formats supported for importing embedded video, see "About file formats for imported video" on page 178.
- You can import video clips in Macromedia Flash Video (FLV) format directly into Flash. When you import FLV files, you can use the encoding options already applied to the files. You do not need to select encoding options during import. For more information, see "Importing Macromedia Flash Video (FLV) files" on page 187.

  Note: You can export video clips in FLV format from Flash, for use in Flash documents or other applications. For more information, see "Macromedia Flash Video (FLV)" on page 349.

- You can play back external FLV files in a Flash document at runtime, using the NetConnection and NetStream objects in ActionScript. For more information, see "About playing back external FLV files dynamically" on page 189.
- You can import video clips in QuickTime format as linked files. Flash documents that contain linked QuickTime video must be published in QuickTime format. A linked video file does not become part of the Flash document. Instead, the Flash document maintains a pointer to the linked file. For more information, see "Importing linked QuickTime video files" on page 188.
- If you have Macromedia Flash MX Professional 2004, you can use the FLV file format to create and import video in a streamlined workflow. You can export FLV files from video-editing applications using the FLV plug-in. For more information, see "Exporting FLV files from video-editing applications (Flash Professional only)" on page 192. You can play back FLV files using the streaming media components. For more information, see "Playing FLV video clips with media components (Flash Professional only)" on page 196.

There are several options for controlling playback of imported video files:

- You can use video behaviors (prewritten ActionScript scripts) to control video playback. For more information, see "Controlling video playback using behaviors" on page 191.

- If you are comfortable with ActionScript, you can write custom ActionScript to control video playback. You can play or stop a video, jump to a frame, and control video in other ways. You can also display a live video stream from a camera. For more information, see "About controlling video playback using the Timeline" on page 192.

Note: You can preview frames of an imported video by dragging the playhead along the Timeline. However, the sound does not play back. To preview the video with sound, use the Test Movie command.

You can use the Property inspector and the Embedded Video Properties dialog box to modify embedded and linked video clips. The Property inspector lets you give the clip an instance name; change the width, height, and registration points; and swap a video clip with another video clip. The Embedded Video Properties dialog box lets you rename a video clip, update an imported video that you have edited in an external application, or import another video to replace the selected clip. For more information, see "Changing the properties of a video clip" on page 190.

For lessons on working with video, see Import and Edit Video on the Macromedia Flash Support Center at www.macromedia.com/support/flash/images_video/flash_video/.

About file formats for imported video

If you have QuickTime 4 or later (Windows or Macintosh) or DirectX 7 or later (Windows only) installed on your system, you can import embedded video clips in several file formats, including MOV, AVI, and MPG/MPEG. You can import linked video clips in MOV format.

Flash documents with embedded video can be published as SWF files. Flash documents with linked video must be published in QuickTime format.

The following video file formats are supported for importing embedded video if QuickTime 4 is installed:

File type	Extension
Audio Video Interleaved	.avi
Digital video	.dv
Motion Picture Experts Group	.mpg, .mpeg
QuickTime video	.mov

The following video file formats are supported for importing embedded video if DirectX 7 or later is installed (Windows only):

File type	Extension
Audio Video Interleaved	.avi
Motion Picture Experts Group	.mpg, .mpeg
Windows Media file	.wmv, .asf

By default, Flash imports and exports video using the Sorenson Spark *codec*. A codec is a compression/decompression algorithm that controls how multimedia files are compressed and decompressed during import and export. For information on the Sorenson Spark codec, see "About the Sorenson Spark codec" on page 179.

If you attempt to import a file format that is not supported on your system, Flash shows a warning message indicating that the operation cannot be completed. In some cases, Flash might import the video but not the audio in a file. For example, audio is not supported in MPG/MPEG files imported with QuickTime 4. In such cases, Flash shows a warning indicating that the audio portion of the file cannot be imported. You can still import the video without sound.

Note: Imported audio is published or exported as streamed audio, using the global audio streaming settings selected in the Publish Settings dialog box. For more information, see "Setting publish options for the Flash SWF file format" on page 312.

About the Sorenson Spark codec

Sorenson Spark is a motion video codec included in Flash that lets you add embedded video content to Flash. Spark is a high-quality video encoder and decoder that dramatically lowers the bandwidth required to deliver video into Flash while simultaneously increasing the video quality. By including Spark, Flash makes important progress in video capability. In Flash 5 or earlier, you could only simulate video using sequential bitmap images.

Two versions of Sorenson Spark are available: Sorenson Spark Standard Edition is included in Flash MX 2004 and Flash Player 7. The Spark Standard edition codec produces good-quality video for low-motion content, such as a person speaking. The Spark video codec comprises an encoder and a decoder. The encoder (or *compressor*) is the component in Spark that compresses your content. The decoder (or *decompressor*) is the component that decompresses the compressed content so that it can be viewed. The decoder is included in Flash Player.

There are two types of compression that can be applied to digital media: *spatial* and *temporal*.

Temporal compression identifies the differences between frames and stores only those differences, so that frames are described based on their difference from the preceding frame. Unchanged areas are simply repeated from the previous frame(s). A temporally compressed frame is often called an *interframe*.

Spatial compression, however, is applied to a single frame of data, independent of any surrounding frames. Spatial compression can be *lossless* (in which no data is discarded from the image) or *lossy* (in which data is selectively discarded). A spatially compressed frame is often called an *intraframe*.

Sorenson Spark is an interframe codec. Sorenson Spark's efficient interframe compression, among other features, distinguishes it from other compression technologies, requiring a much lower data rate than most other codecs to produce good-quality video. Many other codecs use intraframe compression; for example, JPEG is an intraframe codec.

However, interframe codecs also use intraframes. The intraframes are used as the reference frames (keyframes) for the interframes. Sorenson Spark always begins with a keyframe. Each keyframe becomes the main reference frame for the following interframes. Whenever the next frame is significantly different from the previous frame, the codec compresses a new keyframe.

Tips for creating Flash video with Sorenson Spark

How you compress your video is largely determined by the content of the video. A video clip of a talking head with little action and only short bursts of moderate motion compresses differently than footage of a soccer match. The following tips discuss delivering the best possible Flash video:

Strive for simplicity Avoid elaborate transitions—they don't compress well and can make your final compressed video look "chunky" during the change. Hard cuts or quick cross-fades are usually best. Video sequences that show an object zooming from behind the first track, doing a "page turn," or wrapping around a ball and then flying off the screen can be eye-catching, but they usually don't compress well and should be used sparingly.

Know your audience data rate When you deliver video over the Internet, you should produce files at lower data rates. Users with fast Internet connections can view the files with little or no wait, but dialup users must wait for files to download. It is best to make the clips short to keep the download times within acceptable limits for dialup users.

Select the proper frame rate Frame rate indicates how many frames play each second (fps). If you have a higher data rate clip, a lower frame rate can improve playback on lower-end computers. For example, if you are compressing a talking head clip with little motion, cutting the frame rate in half will probably save you only 20% of the data rate. However, if you are compressing high-motion video, reducing the frame rate has a much greater effect on the data rate.

Because video looks much better at native frame rates, Macromedia recommends leaving it high if allowed by your delivery channels and playback platforms. However, if you need to reduce the frame rate, the best results come from dividing the frame rate by whole numbers.

Select a frame size that fits your data rate As with the frame rate, the frame size for your document is important for producing high-quality video. At a given data rate (connection speed), increasing the frame size results in decreased video quality. When you select the frame size for your document, you must consider frame rate, source material, and personal preferences. The following list of common frame sizes can be used as a guideline. You can experiment to find the best setting for your project.

- Modem: 160 x 120
- Dual ISDN: 192 x 144
- T1/DSL/cable: 320 x 240

Know progressive download You should know how long it is going to take to download your video. While your video clip downloads, you might want to have other content that appears and "disguises" the download. For short clips, you can use the following formula: Pause = download time – play time + 10% of play time. For example, If your clip is 30 seconds long and it takes one minute to download, you should give your clip a 33-second buffer: 60 seconds – 30 seconds + 3 seconds = 33 seconds.

Use clean video The higher the quality of the original, the better the final result. Although frame rates and sizes of Internet video are usually smaller than those of television, computer monitors have much better color fidelity, saturation, sharpness, and resolution than conventional televisions. Even with a small window, image quality can be more important for digital video than for standard analog television. Artifacts and noise that would hardly be noticeable on TV can be quite obvious on a computer screen.

Remove noise and interlace After you capture your video content, you might need to remove noise and interlace.

Follow the same guidelines for audio The same considerations exist for audio production as for video production. To achieve good audio compression, you must begin with clean audio. If you are encoding material from a CD, try to record the file using direct digital transfer instead of through the analog input of your sound card. The sound card introduces an unnecessary digital-to-analog and analog-to-digital conversion that can create noise in your source audio. Direct digital transfer tools are available for Windows and Macintosh platforms. If you must record from an analog source, you should use the highest quality sound card available.

Using the Video Import wizard

The Video Import wizard provides a streamlined interface for importing video into a Flash document. The wizard lets you select whether to import a video clip as an embedded or a linked file.

When you import a video clip as an embedded file, you select options in the wizard for encoding and editing the video. Click the Next button to advance through panes in the wizard, and click the Back button to return to previous panes.

You can import video clips as embedded files in several file formats, depending on your system. For information on supported file formats, see "About file formats for imported video" on page 178. You can preview frames of an imported video by dragging the playhead along the Timeline. However, the sound does not play back. To preview the video with sound, use the Test Movie command. For more information, see "Testing document download performance" on page 41

When you import a video as an embedded file, you have the option to edit the video before importing it. You can also apply customized compression settings, including bandwidth or quality settings, as well as advanced settings for color correction, cropping, and other options. You select editing and encoding options in the Video Import wizard. After a video clip is imported, it cannot be edited.

Embedded video is encoded using the Sorenson Spark codec. For more information, see "About the Sorenson Spark codec" on page 179.

Using the Property inspector, you can give an embedded clip an instance name; change its width, height, and position on the Stage; and swap the embedded clip with another video clip. You can use the Embedded Video Properties dialog box to rename a video clip, update an imported video clip that you have edited in an external application, or import another video to replace the selected clip. For more information, see "Changing the properties of a video clip" on page 190.

You can export an embedded video as a FLV file, which retains the compression settings applied when the FLV file was created. For more information, see "Macromedia Flash Video (FLV)" on page 349.

For lessons on working with video, see Help > How Do I > Quick Tasks > Create a Document or Import and Edit Video.

To import an embedded video clip:

1. Do one of the following:
   - To import the video clip directly to the Stage in the current Flash document, select File > Import > Import to Stage.
   - To import the video clip into the library for the current Flash document, select File > Import > Import to Library.
2. Do one of the following:
   - To import the entire video clip without editing it, select Import the Entire Video, and click Next. Proceed to step 3 to continue selecting compression options for the video.
   - To edit the video clip before importing it, select Edit the Video First, and click Next. To select editing options for the video, see "Editing video clips in the Video Import wizard" on page 182.
3. Do one of the following:
   - To apply a predefined Compression Profile, select a Bandwidth option from the pop-up menu.
   - To create a custom compression profile, select Create New Profile or a predefined compression rate from the Compression profile pop-up menu, and click Edit. For more information, see "Editing video clips in the Video Import wizard" on page 182.
4. To apply advanced video encoding to specify color, dimensions, track, and audio options, select Create New Profile from the Advanced Settings pop-up menu. For more information, see "Selecting advanced settings in the Video Import wizard" on page 185.
5. Click Finish to close the Video Import wizard and complete the video import procedure.

To update an embedded video clip after editing it in an external editor:

1. Select the video clip in the Library panel.
2. In the options menu in the upper right corner of the Library panel, select Properties.
3. Click Update in the Embedded Video Properties dialog box.

   The embedded video clip is updated with the edited file. The compression settings you selected when you first imported the video are reapplied to the updated clip.

Editing video clips in the Video Import wizard

The Video Import wizard provides editing options that let you edit embedded video as you import it. You can select in and out points for a clip, create multiple clips from one imported clip, and select other editing options. Editing as you import video clips is especially useful with raw footage.

To edit an embedded video clip:

1. Import an embedded video clip.
2. Select Edit the Video First, and click Next to open the Editing pane of the Video Import wizard.
3. To browse frames in the video, do one of the following:
   - Drag the playhead along the scrubber bar.
   - Click the Play button to move forward and the Pause button to stop at the desired frame.
   - Click the Backward and Forward buttons in the Controller to move forward or backward one frame at a time.
4. To set the in and out points (beginning and ending frames), do one of the following:
   - Drag the in and out points (the triangles below the scrubber bar).
   - Click the In or Out button in the button controls below the scrubber bar to set the beginning or ending frame at the current location of the playhead.
5. To play the video, do one of the following:
   - Click the Play button in the button controls to play the video from the current playhead position.
   - Click Preview to play the video with the current in and out points.

   Note: Click the Stop button in the button controls to stop video playback.
6. To create a clip with the current in and out points, click Create Clip. The clip appears in the scroll pane at the left of the Editing pane.

   To create additional clips from the same file, select in and out points for the clips as described in step 4, and click Create Clip again.
7. To rename a clip, select it in the scroll pane and enter the new name.
8. To re-edit a clip, select it in the scroll pane. Select new in and out points as described in step 4, and click Update Clip.
9. To combine all clips in the scroll pane into a single clip for import, select Combine List of Clips into a Single Library Item After Import.
10. To change the order of clips in the scroll pane, select a clip in the scroll pane and click the Up Arrow or Down Arrow button.

    Note: The order of clips in the scroll pane is the order in which clips appear if you combine them into a single clip for import.
11. To delete a clip from the scroll pane, select the clip and click the Delete (-) button.
12. When you have completed the editing process, click Next to advance to the next pane in the Video Import wizard.

Selecting compression profiles in the Video Import wizard

The Video Import wizard provides several options for compressing a video clip during the import process. In the Encoding panel, you can enter a value for Bandwidth or Quality, control the frequency of keyframes, ensure consistent image quality in keyframes, increase encoding speed, and match the playback speed of the imported video to the playback speed of the main Flash document Timeline.

You select a compression profile to select the level of compression that will be applied to the imported embedded video. You can select a profile based on bandwidth or video quality.

- Custom bandwidth options range from 0 to 750 Kbps and specify the approximate download speed, in kilobits per second, for the video. Preset options include 56 Kbps modem, 256 Kbps, 512 Kbps, and 786 Kbps on DSL or cable. The quality setting of individual frames can vary to achieve a consistent download speed.
- Video quality settings, which range from 0 to 100, specify a compression level for all frames. You can also specify a keyframe rate. The download speed can vary to achieve a consistent compression level.

To reduce the time it takes to compress a file, you can select Quick Compress.

You can synchronize the frame rate of an embedded video to match the frame rate of the main Timeline. You can also adjust the ratio of the video frame rate to the main Timeline frame rate, to drop frames from the imported video during playback.

In some situations, you might not want to synchronize the embedded video with the SFW file. Instead, you want to prevent frames in the embedded video from being dropped or duplicated. For example, suppose you want to import an NTSC video clip with a frame rate of 29.97 fps into a Flash document with a frame rate of 30 fps. Deselecting the Synchronize option keeps frames from being dropped in the embedded video and prevents the hiccup effect that this causes during playback.

You can save customized compression profiles as named settings. The new settings appear in the Compression Profile pop-up menu.

To create a custom compression profile:

1. From the Compression Profile pop-up menu in the Encoding panel, select Create New Profile or a predefined compression rate and click Edit.
2. Do one of the following:
   - Select Bandwidth and drag the slider or enter a Bandwidth value between 0 and 750 Kbps. Bandwidth options specify the approximate download speed, in kilobits per second, for the video. The quality setting of individual frames can vary to achieve a consistent download speed.
   - Select Quality, and drag the slider or enter a quality value between 0 and 100. Quality options specify a compression level for all frames. Higher values produce better images but increase download time. The download speed can vary to achieve a consistent compression level.

3. Drag the slider or enter a value for Keyframe Interval to control the frequency of keyframes (frames with complete data) in the video clip. For example, with a keyframe interval of 30, Flash stores a complete frame every 30 frames. For frames between intervals, Flash stores only the data that changes from the preceding frame. With smaller intervals, you can fast-forward or rewind more quickly to a specific frame, but the file size is larger.

   Note: A keyframe interval of 1 stores a complete frame for each frame of the video. This setting is recommended only for very small video files.

4. If you are encoding the video using a bandwidth value, select High Quality Keyframes to ensure consistent image quality in keyframes. Using a consistent bandwidth speed can reduce the quality of keyframes if you do not select this option.
5. Select Quick Compress to reduce the time it takes to compress a file. Increasing encoding speed can also decrease image quality.
6. Select Synchronize Video to Macromedia Flash Document Frame Rate to match the playback speed of the imported video to the playback speed of the main Flash document Timeline. Deselect this option to prevent frame rate synchronization.
7. Select a value for Number of Video Frames to Encode Per Number of Flash Frames to specify the ratio of imported video frames to main Flash Timeline frames. For example, to play one imported video frame for every main Flash Timeline frame, select 1:1; to play one imported video frame for every two main Timeline frames, select 1:2.

   Dropping frames from the imported video does not slow down the motion of the video. Instead, it shows fewer frames per second, so that the video appears choppy during playback.
8. Click Next.
9. Enter a name and description on the Encoding (Part 3, Save) panel. Click Next to save the setting.
10. Do one of the following:
    - Edit the Advanced settings.
    - Click Finish.
11. If you selected Current Timeline or Graphic Symbol for Track Options, a notification appears if the imported clip contains more frames than the current Timeline. Do one of the following:
    - Click Yes to add the required number of frames to the current Timeline span.
    - Click No to keep the span at its current size. Frames in the imported clip that exceed the frames in the span do not appear unless you subsequently add frames to the span.

Selecting advanced settings in the Video Import wizard

In the Video Import wizard, you can apply advanced settings to imported videos. Color-correction options let you adjust hue, saturation, brightness, contrast, and gamma to control color quality. Dimensions options let you reduce the scale of the imported video or crop the video from the top, bottom, left, or right edge.

Track options let you select the type of object of the imported video: a video object on the current Timeline, a movie clip on the first frame of the Flash document, or a graphic symbol on the current Timeline. Audio options let you import an audio track as a separate file or an integrated part of the video file, or you can exclude the audio track from the imports.

You can save customized advanced settings as named profiles. The new settings appear in the Advanced Settings pop-up menu.

To apply advanced video encoding settings:

1. In the Video Import wizard, after you specified options for Compression Profile, select Create New Profile from the Advanced Settings pop-up menu. (If you have previously created Advanced Settings profiles, you can select a named setting from the pop-up menu.)
2. Under Color options, enter values or drag the pop-up sliders to apply color corrections to the video image:

   Hue measures the color value, commonly indicated by the color name, such as red or green. Hue is identified as a location on a standard color wheel. Hue value can be between -180° and 180°.

   Saturation measures the strength or purity of the color. Saturation measures the amount of gray in proportion to the hue, which is indicated as a percentage between -100 and +100. A smaller saturation value indicates more gray. A higher value adds more color.

   Brightness measures the relative lightness or darkness of the color, indicated by a percentage between -100 and +100. A smaller value indicates more black, and a larger value indicates more white.

   Contrast measures the contrast between dark and light in the image, indicated by a percentage value between -100 and +100. A smaller value indicates less contrast.

   Gamma measures the overall lightness levels, indicated by a value between 0.1 and 1.8. A smaller value indicates a darker image. With a larger value, dark elements in the image stay dark and light elements become lighter.

   Reset resets all Color options to their default values.
3. Under Dimensions options, enter values or drag the pop-up sliders to adjust the following video dimensions:
   - For Scale, enter a value between 0 and 100 to decrease the scale of the video. The Width and Height values indicate the size of the video in pixels. (You cannot increase the scale of the video beyond its original size.)
   - For Crop, enter values for the right, left, top, and bottom edges to crop the video. Guides in the preview window indicate the cropped area.

4. Under Track options, select an option for Import Into to specify the type of object for the imported video:

   Current Timeline imports the video as a video object in the current Timeline in the Flash document. If there are not enough frames in the current Timeline to accommodate the video, Flash prompts you to add more frames on import. With this option, you can browse the video frames in the current Timeline. However, you cannot apply effects to the video object.

   Movie Clip imports the video as a movie clip in the first frame of the Flash document. With this option, you can apply effects. However, you cannot browse the video frames in the current Timeline. (To browse the frames, you must open the Timeline of the movie clip.)

   Graphic Symbol imports the video as a graphic symbol in the current Timeline. If there are not enough frames in the current Timeline to accommodate the video, Flash prompts you to add more frames on import. With this option, you can browse the video frames in the current Timeline, and you can apply effects to the video.

5. For Audio Track, select one of the following options to specify how audio can be imported:

   Separate imports the audio track as a sound object, separate from the video file.

   Integrated imports the audio track as part of the video file.

   None does not import the audio track.

6. Click Next.

7. Enter a name and description for the Advanced Settings to save the settings. The name appears in the Advanced Settings pop-up menu the next time you use the Video Import wizard. Click Next.

8. Click Finish to close the Video Import wizard and import the video.

9. If you selected Current Timeline or Graphic Symbol for Track Options, a warning appears if the imported clip contains more frames than the current Timeline. Do one of the following:

   - Click Yes to add the required number of frames to the current Timeline span.
   - Click No to keep the span at its current size. Frames in the imported clip that exceed the frames in the span do not appear unless you subsequently add frames to the span.

Importing Macromedia Flash Video (FLV) files

The FLV file format lets you import or export a static video stream with encoded audio. This format can be used with communications applications, such as video conferencing.

Files in the FLV format are compressed with the Sorensen codec. For more information, see "About the Sorenson Spark codec" on page 179.

You can import files in the FLV format using the Import or Import to Library commands or the Import button in the Embedded Video Properties dialog box.

To import a video clip in FLV format, do one of the following:

- Select File > Import or File > Import to Library.
- Select any existing video clip in the Library panel, and select Properties from the Library options menu. In the Embedded Video Properties dialog box, click Import. Locate the file you want to import, and click Open in the Open dialog box.

Importing linked QuickTime video files

If you are importing a QuickTime video clip, you can link to the video from the Flash file, rather than embed the video. A linked QuickTime video imported into Flash does not become part of the Flash file. Instead, Flash maintains a pointer to the source file.

If you link to a QuickTime video, you must publish the SWF file as a QuickTime video. You cannot display a linked QuickTime in SWF format. The QuickTime contains a Flash track, but the linked video clip remains in QuickTime format.

For more information on publishing your Flash file as a QuickTime video, see "Specifying publish settings for QuickTime videos" on page 325.

You can scale, rotate, and animate a linked QuickTime video in Flash. However, you cannot tween linked QuickTime video content in Flash.

Note: The QuickTime Player does not currently support Flash Player 6 files. For more information, see "Specifying publish settings for QuickTime videos" on page 325.

To import a QuickTime video as a linked file:

1. Do one of the following:
   - To link the video clip directly to the current Flash document, select File > Import > Import to Stage.
   - To link the video clip to the library for the current Flash document, select File > Import > Import to Library.
2. In the Import Video wizard, select Link to External Video File, and click Next.
3. If you imported the video clip directly to the Stage in step 1, a warning appears if the imported clip contains more frames than the span in which you are placing it in the current Flash document. Do one of the following:
   - Click Yes to extend the span the required number of frames.
   - Click No to keep the span at its current size. Frames in the imported clip that exceed the frames in the span do not appear unless you subsequently add frames to the span.

You can preview a linked QuickTime video before you publish your SWF file. When you import a linked QuickTime video, Flash adds the required number of frames to preview the QuickTime video, the same as it does for an embedded video.

Note: You cannot preview linked QuickTime video content using the Test Movie command.

To preview a linked QuickTime video:

- Select Control > Play.

Setting the directory path of a linked QuickTime video

You can set the directory path of a linked QuickTime video clip in the library for the current Flash document.

To set the directory path of a linked QuickTime video clip:

1. Select Window > Library, and select the desired linked QuickTime video.
2. In the options menu in the upper right corner of the Library panel, select Properties.
3. Click Set Path in the Linked Video Properties dialog box.
4. In the Open dialog box, navigate to the file for the linked video clip and select it, then click Open.
5. In the Linked Video Properties dialog box, click OK.

About playing back external FLV files dynamically

As an alternative to importing video into the Flash authoring environment, you can use ActionScript to dynamically play external FLV files in Flash Player. You can play FLV files posted as HTTP downloads or as local media files. To play FLV files, you use the NetStream object and the `attachVideo` method of the Video object.

You can create FLV files by importing video into the Flash authoring tool and exporting it as an FLV file. For information on exporting video as an FLV file, see "Macromedia Flash Video (FLV)" on page 349. If you have Macromedia Flash MX Professional 2004, you can use the FLV Export plug-in to export FLV files from supported video-editing applications. For more information, see "Exporting FLV files from video-editing applications (Flash Professional only)" on page 192.

To play back an external FLV file, you must post an FLV file to a URL (either an HTTP site or a local folder) and add ActionScript code to the Flash document to access the file and control playback during runtime.

Using external FLV files provides certain capabilities that are not available when using imported video, as described in the following list:

- You can use longer video clips in your Flash documents without slowing down playback. External FLV files are played using *cached memory*, which means that large files are stored in small pieces and accessed dynamically; they do not require as much memory as embedded video files.
- An external FLV file can have a different frame rate from the Flash document in which it plays. For example, you can set the Flash document frame rate to 30 fps and the video frame rate to 21 fps, which gives you greater control in ensuring smooth video playback.
- With external FLV files, Flash document playback does not have to be interrupted while the video file is loading. Imported video files can sometimes interrupt document playback to perform certain functions (for example, to access a CD-ROM drive). FLV files can perform functions independently of the Flash document and, so, do not interrupt playback.
- Captioning video content is easier with external FLV files because you can use callback functions to access metadata for the video.

For more information on playing back FLV files, see "Playing back external FLV files dynamically" in *Using ActionScript in Flash*.

Changing the properties of a video clip

You can use the Property inspector to change properties for an instance of an embedded or linked video clip on the Stage. In the Property inspector, you can assign the instance an instance name and change its width, height, and position on the Stage. You can also *swap* an instance of a video clip—assign a different symbol to an instance of a video clip. Assigning a different symbol to an instance displays a different instance on the Stage but leaves all the other instance properties (such as dimensions and registration point) intact.

The Embedded Video Properties dialog box lets you view information about an imported video clip, including its name, path, creation date, pixel dimensions, length, and file size. You can change the video clip name, update the video clip if you modify it in an external editor, and import an FLV video to replace the selected clip.

Note: You can also export a video clip as an FLV file using the Embedded Video Properties dialog box. For more information, see "Macromedia Flash Video (FLV)" on page 349.

To change video instance properties in the Property inspector:

1. Select an instance of an embedded or linked video clip on the Stage.
2. Select Window > Properties.
3. In the Property inspector, do any of the following:

- Enter an instance name in the Name text box on the left side of the Property inspector.
- Enter values for W and H to change the dimensions of the video instance.
- Enter values for X and Y to change the position of the upper left corner of the instance on the Stage.
- Click Swap. In the Swap Embedded Video dialog box, select a video clip to replace the one currently assigned to the instance.

  Note: You can swap an embedded video clip only with another embedded video clip, and you can swap a linked video clip only with another linked video clip.

To view video clip properties in the Embedded Video Properties dialog box:

1. Select a video clip in the Library panel.
2. Select Properties from the Library options menu.

To assign a new name to a video clip:

1. Select the video clip in the Library panel.
2. Select Properties from the Library options menu.
3. In the Embedded Video Properties dialog box, enter a new name in the Name text box.

To update a video clip:

1. Select the video clip in the Library panel.
2. Select Properties from the Library options menu.
3. In the Embedded Video Properties dialog box, click Update.
4. Navigate to the updated video file and click Open. The file is reimported into the Flash document.

To replace a video clip with an FLV clip:

1. Select the video clip in the Library panel.
2. Select Properties from the Library options menu.
3. In the Embedded Video Properties dialog box, click Import.
4. Navigate to the FLV file that will replace the current clip, and click Open.

Controlling video playback using behaviors

Video behaviors provide one way to control video playback. Behaviors are prewritten ActionScript scripts that you add to a triggering object to control another object. Behaviors let you add the power, control, and flexibility of ActionScript coding to your document without having to create the ActionScript code. Video behaviors let you play, stop, pause, rewind, fast-forward, show, and hide a video clip.

To control a video clip with a behavior, you use the Behaviors panel to apply the behavior to a triggering object, such as a movie clip. You specify the event that will trigger the behavior (such as releasing the movie clip), select a target object (the video that is affected by the behavior), and when necessary, select settings for the behavior, such as the number of frames to rewind.

The triggering object must be a movie clip. It is not possible to attach video playback behaviors to button symbols or button components.

The following behaviors come with Flash MX 2004 and Flash MX Professional 2004 and are used to control embedded video:

Behavior	Purpose	Parameters
Play Video	Plays a video in the current document.	Instance name of target video
Stop Video	Stops the video.	Instance name of target video
Pause Video	Pauses the video.	Instance name of target video
Rewind Video	Rewinds the video by the specified number of frames.	Instance name of target video Number of frames
Fast Forward Video	Fast-forwards the video by the specified number of frames.	Instance name of target video Number of frames
Hide Video	Hides the video.	Instance name of target video
Show Video	Shows the video.	Instance name of target video

For lessons on working with video, select Help > How Do I > Quick Tasks > Create a Document or Import and Edit Video.

To add and configure a behavior:

1. Select the movie clip that will trigger the behavior.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button and select the desired behavior from the Embedded Video submenu.
3. In the dialog box that appears, select the video you want to control with the behavior.
4. Select a Relative or Absolute path. For more information, see "Using absolute and relative target paths" on page 23.
5. If required, select settings for the behavior parameters and click OK.

   Default event and actions for the behavior appear in the Behaviors panel.
6. In the Behaviors panel, under Event click On Release (the default event) and select a mouse event from the menu. If you want to use the On Release event, leave the option unchanged.

About controlling video playback using the Timeline

You can control playback of an embedded or linked video file by controlling the Timeline that contains the video. For example, to pause a video playing on the main Timeline, you would call a `stop()` action that targets that Timeline. Similarly, you can control a video object in a movie clip symbol by controlling the playback of that symbol's Timeline.

You can apply the following actions to imported video objects in movie clips: `goTo`, `play`, `stop`, `toggleHighQuality`, `stopAllSounds`, `getURL`, `FScommand`, `loadMovie`, `unloadMovie`, `ifFrameLoaded`, and `onMouseEvent`. To apply actions to a Video object, you must first convert the Video object to a movie clip. For more information, see "Video class" in *Flash ActionScript Language Reference*.

You can also use ActionScript to show a live video stream from a camera. First, use the New Video Object in the Library panel to place a Video object on the Stage. Then use `Video.attachVideo` to attach the video stream to the Video object. For more information, see `Video.attachVideo()` in *Flash ActionScript Language Reference*.

Exporting FLV files from video-editing applications (Flash Professional only)

If you have Macromedia Flash MX Professional 2004 and QuickTime 6.1.1 installed on your computer, you can use the FLV Export plug-in to export FLV files from supported video-editing applications. You can then import these FLV files directly into Flash to use in your Flash documents. In addition, you can dynamically play back external FLV files in Flash documents at runtime. See "About playing back external FLV files dynamically" on page 189.

Exporting FLV files from video-editing applications significantly streamlines the workflow in using FLV files in your Flash documents. With the FLV Export plug-in, you can select encoding options for video and audio content as you export, including frame rate, bit rate, quality, and other options. You can import FLV files directly into Flash without needing to re-encode the video after import. For more information on video formats and importing to Flash, see "Video Fundamentals" at www.macromedia.com/support/flash/.

The following video-editing applications are supported by the FLV Export plug-in:

- Adobe After Effects (Windows and Macintosh)
- Anystream Agility (Windows)
- Apple FinalCut Pro (Macintosh)
- Apple QuickTime Pro (Macintosh)
- Avid Xpress DV (Windows and Macintosh)
- Discreet Cleaner (Windows and Macintosh)
- Discreet Cleaner XL (Windows and Macintosh)

You can install the FLV Export plug-in after installation of Flash MX Professional 2004 is complete. The plug-in is placed in the QuickTime folder. From this location, the FLV Export plug-in is available to any application that uses QuickTime 6.1.1, including programs added after the FLV installation.

To install the FLV Export plug-in:

- Click the installer, and follow the instructions.

When you export video from an editing application, the Flash Video (FLV) Exporter is used to set various encoding options. The Flash Video Exporter is divided into three main sections: Video, Audio, and Other. The Video section contains the encoding options for your FLV video, Audio contains the bit-rate options for your MP3 audio encoding, and Other lets you set your scaling and deinterlacing options.

To export an FLV file from a supported application:

1. With the video open in the video-editing application, select File > Export > QuickTime.
2. In the Export dialog box, under Export, select Macromedia Flash Video (FLV), and click Options.

3. In the Flash Video (FLV) Exporter dialog box, select an Encoding Method from the pop-up menu:

   Baseline (1 Pass) is the most basic method of encoding. It reads through the file once, compresses each frame as it is loaded, and estimates the bit rate for each frame. The resulting file size is closely related to the value entered in the Limit Data Rate To text box because the data rate can not vary on a frame-by-frame basis, even if a different amount of data is needed.

   Better (1 Pass VBR) is the same as Baseline but encodes using Variable Bitrate Encoding (VBR). VBR keeps the average bit rate at or below the target data rate specified for the video. This lets the data rate fluctuate based on the content in the clip. If VBR can encode the frame with less data, it will. The resulting file looks similar to one created with the Baseline option, but the file size can be smaller.

   Screen Recording Codec is for recording screen operations with a lossless compression. This compression optimizes content that depicts computer screens; for example, an instructional video that shows a pointer moving across a computer screen.When Screen Recording Codec is selected, Quality, Limit Data Rate To, and Motion Estimation are dimmed.

4. For Frames per Second, enter a value or select a frame rate from the pop-up menu. To maintain the temporal quality of the original source clip, use the same frame rate as that of the original source. If you are dropping the frame rate to lower the data rate, the frames per second should be an evenly divisible number from the source frame rate (that is, one-half or one-quarter the original rate). The pop-up menu next to the Frames per Second text box contains commonly used frame rates.

5. For Limit Data Rate To, do one of the following:
   - Select a preset Quality setting (Normal, Better, or Best) to automatically select a Limit Data Rate value. This value is determined by the output FLV file's resolution and frame rate. When you select Low, Medium, and High from this menu, the Limit Data Rate To text box updates to reflect the specified value.
   - Use the pop-up window to select a value for Kilobits/Sec.
   - If you find that the preset quality settings are not working with your particular source footage, enter a higher data rate in the Limit Data Rate To text box.
6. For Keyframe, enter a value to control the frequency of keyframes (frames with complete data) in the video clip. For example, for a keyframe interval of 30, Flash stores a complete frame every 30 frames in the clip. For frames between intervals, Flash stores only the data that changes from the preceding frame. A smaller interval stores more complete frames, resulting in a larger file size.
7. For Motion Estimation, select Faster for faster encoding and lower video quality, or Best for slower encoding and better video quality.
8. For Audio Encoding, select from the following options:
   - Select Audio to export audio content with the video file. Deselect this option to export no audio with the file.
   - Select Mono to export all audio content in one channel, or Stereo to export audio content in stereo.
   - For Bitrate, specify the maximum bit rate at which the audio content will be transmitted. (Audio is encoded separately from the video content in the file and can have a different bit rate.)
9. Under Other, do one of the following to resize the video:
   - Select a preset image size from the pop-up menu.
   - Specify values for Width and Height, in pixels.
   - Specify a percentage of the original image size.
10. Check Lock Aspect Ratio to keep the aspect ratio the same size as the original clip.
11. For Deinterlacing, select None to apply no deinterlacing, Lower to deinterlace for NTSC format, or Upper to deinterlace for PAL format. This option lets you choose whether to strip the Upper or Lower field from the interlaced source video file.
12. Click OK.
13. Select a location to save the exported file, and click Save.

Playing FLV video clips with media components (Flash Professional only)

With Flash MX Professional 2004 media components, you can quickly and easily add Flash video and playback controls to your documents. Then, using cue points, you can synchronize your video with animation, text, and graphics. For example, you can create a Flash presentation that has video playing in one area of the screen while text and graphics appear in another area. A cue point placed in the video triggers an update to the text and graphic, letting them remain relevant to the content of the video.

The media component suite consists of three components: MediaDisplay, MediaController, and MediaPlayback. With the MediaDisplay component, adding media to your Flash documents is as simple as dragging the component to the Stage and configuring it in the Component inspector. In addition to setting the parameters in the Component inspector, you can add cue points to trigger other actions. The MediaDisplay component has no visual representation during playback; only the video clip is visible. For more information, see "Media components (Flash Professional only)" in *Using Components*.

The MediaController component provides user interface controls that let the user interact with streaming media. The Controller features Play, Pause, and Rewind to Start buttons and a volume control. It also includes playbars that show how much of the media has loaded and how much has played. A playhead slider can be dragged forward and backward on the playbar to navigate quickly to different parts of the video. Using Behaviors or ActionScript, you can easily link this component to the MediaDisplay component to show streaming video and provide user control. For more information, see "Media components (Flash Professional only)"in *Using Components*.

The MediaPlayback component provides the easiest and quickest way to add video and a controller to your Flash documents. The MediaPlayback component combines the MediaDisplay and MediaController components into a single, integrated component. The MediaDisplay and MediaController component instances are automatically linked to each other for playback control. For more information, see "Media components (Flash Professional only)" in *Using Components*.

You use the Component inspector to configure parameters for playback, size, and layout for all three components. All the media components work equally well with MP3 audio content.

To add a MediaPlayback component to a Flash document:

1. Open the Components panel (Windows > Development Panels > Components) and drag the MediaPlayback component to the Stage. For more information, see "Adding components to Flash documents" in *Using Components*.
2. With the component selected, open the Property inspector (Windows > Properties) and enter an instance name.
3. Open the Component inspector (Windows > Development Panels > Component Inspector) and select FLV (default setting) for the media type.
4. Enter values for parameters or use default settings:

   Video Length determines progress of the playback by the playbar component.

   Milliseconds determines whether the playbar and cue points use frames or milliseconds.

fps sets the number of frames per second for video playback. When Milliseconds is selected, the frames per second control is disabled.

URL sets the path and filename or URL for the media.

Automatically Play sets the media to play as soon as it is available.

Use Preferred Media Size displays the FLV video clip at its native size and aspect ratio. When deselected, the media conforms to the height and width set in the Component inspector.

Respect Aspect Ratio retains the original aspect ratio of the media when selected.

Control Placement determines if the controller can sit above, below, to the right, or to the left of the video clip.

Control Visibility determines whether the Controller opens or closes based on mouse position or is locked in the open or closed state.

To add a MediaDisplay component to a Flash document:

1. Open the Components panel (Windows > Development Panels > Components), and drag the MediaDisplay component to the Stage. For more information, see "Adding components to Flash documents" in *Using Components*.
2. With the component selected, open the Property inspector (Windows > Properties), and enter an instance name.
3. Open the Component inspector (Windows > Development Panels > Component Inspector), and select FLV (Default setting) for the media type.
4. Enter values for parameters or use default settings:

   Video Length determines progress of the playback by the playbar component.

   Milliseconds determines whether the playbar and cue points use frames or milliseconds.

   fps sets the number of frames per second for video playback. When Milliseconds is selected, the frames per second control is disabled.

   URL sets the path and filename or URL for the media.

   Automatically Play sets the media to play as soon as it is available.

   Use Preferred Media Size displays the FLV video clip at its native size and aspect ratio. When deselected, the media conforms to the height and width set in the component inspector.

   Respect Aspect Ratio retains the original aspect ratio of the media when selected.

To add a MediaController to a Flash document:

1. Open the Components panel (Windows > Development Panels > Components), and drag the MediaController component to the Stage. For more information, see "Adding components to Flash documents" in *Using Components*.
2. With the component selected, open the Property inspector (Windows > Properties), and enter an instance name for the component.

3. Open the Component inspector set the following parameters:

   ActivePlayControl sets the playbar in Play or Pause when the SWF file opens. Use this parameter with Automatically Play in the MediaDisplay component.

   BackgroundStyle indicates whether the background of the controller appears as default or as none.

   ControllerPolicy determines whether the Controller opens or closes based on mouse position or is locked in the open or closed state.

   Horizontal determines whether the Controller orientation is vertical or horizontal.

   Enabled lets the user access the playback controls.

   Visible lets the user see the Controller.

   MinHeight sets the minimum height (in pixels) allowable for this instance.

   MinWidth sets the minimum width (in pixels) allowable for this instance.

The media components use events to interact with other elements in a Flash document, including each other. The MediaController instance broadcasts events when its buttons are clicked or its sliders are dragged. The MediaDisplay instance broadcasts events when playback starts and finishes, the playhead moves, media is downloaded from the source, and when cue points are passed by the playhead.

In order for the MediaController and MediaDisplay instances to work together, they must listen for events from each other and respond appropriately. For example, when a user clicks the Pause button on the MediaController, it broadcasts a "click" event with a detail property of "pause." When the MediaDisplay instance receives the event, it responds by pausing playback.

Flash MX Professional 2004 includes two behaviors, Associate Controller and Associate Display, that connect the MediaDisplay and MediaController components instances. Only one of the two behaviors is necessary to link the two components. The end result is identical with either behavior.

To link a MediaDisplay instance to a MediaController instance:

1. With a MediaDisplay instance and a MediaController instance added to your document (for more information, see "To add a MediaDisplay component to a Flash document:" on page 197 and "To add a MediaController to a Flash document:" on page 197), select the MediaDisplay instance.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button and select the Associate Controller behavior from the Media submenu.
3. In the Associate Controller dialog box, browse to the location of the MediaController instance and select it. If you have not named the instance, you will be asked to enter a name; click OK.

   The behavior inserts the code that enables the component instances to listen to each other.

Defining cue points (Flash Professional only)

A cue point triggers an action when the playhead position equals the value entered in the Position fields. Each cue point consists of a name and the time at which it occurs. By default, cue point times are specified in "hour : minute : second : frame" format, with a default frame rate of 30 fps and can be set to any frame rate. Cue points work with milliseconds or frame numbers.

Cue points also can be added and removed through ActionScript with the `addCuePoint ()` and `removeCuePoint ()` methods. For more information, see "Media components (Flash Professional only)" in *Using Components*.

To add a cue point to a MediaDisplay instance:

1. Select the MediaDisplay instance.
2. In the Component inspector (Window > Development Panels > Component Inspector), click the Add (+) button in the Cue Point panel.
3. Enter the name of the frame or slide, and the time the action is to be triggered.

Adding actions to a cue point (Flash Professional only

Flash MX Professional 2004 provides two cue point behaviors for adding actions to a Flash document, Labeled Frame CuePoint Navigation and Slide CuePoint Navigation. The Labeled Frame CuePoint Navigation behavior adds an action that instructs the Timeline to navigate to a frame with the same name as a given cue point. The Slide CuePoint Navigation behavior instructs a slide-based Flash document to navigate to a slide with the same name as a given cue point and time.

To add a Labeled Frame CuePoint Navigation behavior:

1. Add a blank keyframe (Insert > Timeline > Blank Keyframe) on the same Timeline as the MediaDisplay or MediaPlayback component, and enter a frame label in the Property inspector.
2. Select the MediaDisplay or MediaPlayback component that will trigger the action.
3. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button and select the Labeled Frame CuePoint Navigation behavior from the Media submenu.
4. Select the timeline where your frame label reside (in most cases _root) in the Labeled Frame CuePoint Navigation dialog box.
5. Select Relative, and click OK.

   When the video plays for the amount of time indicated with the cue point, the Flash document navigates to the frame label entered in the cue point.

To add a Slide CuePoint Navigation behavior:

1. Create a slide presentation and name each screen. For more information, see Chapter 12, "Working with Screens (Flash Professional Only)," on page 215.
2. Select the MediaDisplay or MediaPlayback component that will trigger the action.
3. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button and select the Slide CuePoint Navigation behavior from the Media submenu.
4. Select the main slide of your presentation in the Slide CuePoint Navigation dialog box.
5. Select Relative, and click OK.

   When the video plays for the amount of time indicated with the CuePoint, the Flash document navigates to the screen entered in the CuePoint.

CHAPTER 11
Working with Sound

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 offer several ways to use sounds. You can make sounds that play continuously, independent of the Timeline, or you can synchronize animation to a sound track. You can add sounds to buttons to make them more interactive and make sounds fade in and out for a more polished sound track.

There are two types of sounds in Flash: event sounds and stream sounds. An event sound must download completely before it begins playing, and it continues playing until explicitly stopped. Stream sounds begin playing as soon as enough data for the first few frames has been downloaded; stream sounds are synchronized to the Timeline for playing on a website.

You select compression options to control the quality and size of sounds in exported SWF files. You can select compression options for individual sounds using the Sound Properties dialog box or define settings for all sounds in the document in the Publish Settings dialog box.

You can use sounds in shared libraries to link a sound from one library to multiple documents. For more information, see "Using shared library assets" on page 69. You can also use the ActionScript `onSoundComplete` event to trigger an event based on the completion of a sound. For more information, see "About the onSoundComplete event" on page 208.

You can use behaviors that are prewritten ActionScript scripts to load and control the playback of sounds. As with behaviors, the media components contain prewritten ActionScript scripts to load and control sounds (MP3 sounds only) but also provide a controller for stop, pause, rewind, and so on. For more information on using the media components, see "Playing FLV video clips with media components (Flash Professional only)" on page 196.

Note: You can also use actions to load sounds dynamically. For more information, see `Sound.attachSound()` and `Sound.loadSound()` in *Flash ActionScript Language Reference*.

This chapter contains the following sections:

Importing sounds

You place sound files into Flash by importing them into the library for the current document.

Note: When placing a sound on the Timeline, you place it on a separate layer. For more information, see "Adding sounds to a document" on page 203.

You can import the following sound file formats into Flash:

- WAV (Windows only)
- AIFF (Macintosh only)
- MP3 (Windows or Macintosh)

If you have QuickTime 4 or later installed on your system, you can import these additional sound file formats:

- AIFF (Windows or Macintosh)
- Sound Designer II (Macintosh only)
- Sound Only QuickTime Movies (Windows or Macintosh)
- Sun AU (Windows or Macintosh)
- System 7 Sounds (Macintosh only)
- WAV (Windows or Macintosh)

Flash stores sounds in the library along with bitmaps and symbols. As with graphic symbols, you need only one copy of a sound file to use that sound multiple ways in your document.

If you want to share sounds among Flash documents, you can include the sounds in shared libraries. For more information, see "Working with common libraries" on page 21. To use a sound in a shared library, you assign the sound file an identifier string in the Linkage Properties dialog box. The identifier can also be used to access the sound as an object in ActionScript. For information on objects in ActionScript, see "Using sounds with Sound objects" on page 205.

Sounds can use considerable amounts of disk space and RAM. However, MP3 sound data is compressed and smaller than WAV or AIFF sound data. Generally, when using WAV or AIFF files, it's best to use 16-bit 22 kHz mono sounds (stereo uses twice as much data as mono), but Flash can import either 8- or 16-bit sounds at sample rates of 11, 22, or 44 kHz. Flash can convert sounds to lower sample rates on export. For more information, see "Compressing sounds for export" on page 209.

Note: Sounds recorded in formats that are not multiples of 11 kHz (such as 8, 32, or 96 kHz) are resampled when imported into Flash.

If you want to add effects to sounds in Flash, it's best to import 16-bit sounds. If you have limited RAM, keep your sound clips short or work with 8-bit sounds instead of 16-bit sounds.

To import a sound:

1. Select File > Import > Import to Library.
2. In the Import dialog box, locate and open the desired sound file.

   Note: You can also drag a sound from a common library into the library for the current document. For more information, see "Working with common libraries" on page 21.

Adding sounds to a document

To add a sound to a document from the library, you assign the sound to a layer and set options in the Sound controls in the Property inspector. It is recommended that you place each sound on a separate layer.

You can load a sound into a SWF file during runtime, using the `loadSound` method of the Sound object. For more information, see `Sound.loadSound()` in *Flash ActionScript Language Reference*.

To test sounds that you add to a document, you can use the same methods you use to preview frames or test SWF files: Drag the playhead over the frames containing the sound or use commands in the Controller or the Control menu.

To add a sound to a document:

1. Import the sound into the library if it has not already been imported. For more information, see "Importing sounds" on page 202.
2. Select Insert > Timeline > Layer to create a layer for the sound.
3. With the new sound layer selected, drag the sound from the Library panel onto the Stage. The sound is added to the current layer.

   You can place multiple sounds on one layer or on layers containing other objects. However, it is recommended that each sound be placed on a separate layer. Each layer acts as a separate sound channel. The sounds on all layers are combined when you play the SWF file.
4. In the Timeline, select the first frame that contains the sound file.
5. Select Window > Properties and click the arrow in the lower right corner to expand the Property inspector.
6. In the Property inspector, select the sound file from the Sound pop-up menu.
7. Select an effect option from the Effects pop-up menu:

   None applies no effects to the sound file. Select this option to remove previously applied effects.

   Left Channel/Right Channel plays sound in the left or right channel only.

   Fade Left to Right/Fade Right to Left shifts the sound from one channel to the other.

   Fade In gradually increases the volume of a sound over its duration.

Fade Out gradually decreases the volume of a sound over its duration.

Custom lets you create custom in and out points of sound using the Edit Envelope. For more information, see "Using the sound-editing controls" on page 206.

8. Select a synchronization option from the Sync pop-up menu:

   Note: If you are placing the sound on a frame other than Frame 1 in the main TImeline, select the Stop option.

   Event synchronizes the sound to the occurrence of an event. An event sound plays when its starting keyframe first appears and plays in its entirety, independently of the Timeline, even if the SWF file stops playing. Event sounds are mixed when you play your published SWF file.

   An example of an event sound is a sound that plays when a user clicks a button. If an event sound is playing and the sound is instantiated again (for example, by the user clicking the button again) the first instance of the sound continues to play and another instance begins to play simultaneously.

   Start is the same as Event, except that if the sound is already playing, no new instance of the sound plays.

   Stop silences the specified sound.

   Stream synchronizes the sound for playing on a website. Flash forces animation to keep pace with stream sounds. If Flash can't draw animation frames quickly enough, it skips frames. Unlike event sounds, stream sounds stop if the SWF file stops playing. Also, a stream sound can never play longer than the length of the frames it occupies. Stream sounds are mixed when you publish your SWF file.

   An example of a stream sound is the voice of a character in an animation that plays in multiple frames.

   Note: If you use an MP3 sound as a stream sound, you must recompress the sound for export. You can export the sound as an MP3 file, with the same compression settings that it had on import. For more information, see "Compressing sounds for export" on page 209.

9. Enter a value for Repeat to specify the number of times the sound should loop, or select Loop to repeat the sound continuously.

   For continuous play, enter a number large enough to play the sound for an extended duration. For example, to loop a 15-second sound for 15 minutes, enter 60. Looping stream sounds is not recommended. If a stream sound is set to loop, frames are added to the file and the file size is increased by the number of times the sound is looped.

Adding sounds to buttons

You can associate sounds with the different states of a button symbol. Because the sounds are stored with the symbol, they work for all instances of the symbol.

To add sound to a button:

1. Select the button in the Library panel.
2. Select Edit from the options menu in the upper right corner of the panel.
3. In the button's Timeline, add a layer for sound.

4. In the sound layer, create a regular or blank keyframe to correspond with the button state to which you want to add a sound.

   For example, to add a sound that plays when you click the button, create a keyframe in the frame labeled Down.

5. Click the keyframe you created.
6. Select Window > Properties.
7. In the Property inspector, select a sound file from the Sound pop-up menu.
8. Select Event from the Synchronization pop-up menu.

   To associate a different sound with each of the button's keyframes, create a blank keyframe and add another sound file for each keyframe. You can also use the same sound file and apply a different sound effect for each button keyframe. For more information, see "Using the sound-editing controls" on page 206.

Using sounds with Sound objects

You can use the Sound object in ActionScript to add sounds to a document and to control sound objects in a document. Controlling sounds includes adjusting the volume or the right and left balance while a sound plays. For more information, see "Creating sound controls" in *Using ActionScript in Flash*.

To use a sound in a Sound action, you assign an identifier string to the sound in the Linkage Properties dialog box.

To assign an identifier string to a sound:

1. Select the sound in the Library panel.
2. Do one of the following:
   - Select Linkage from the options menu in the upper right corner of the panel.
   - Right-click (Windows) or Control-click (Macintosh) the sound name in the Library panel, and select Linkage from the context menu.
3. Under Linkage in the Linkage Properties dialog box, select Export for ActionScript.
4. Enter an identifier string in the text box, and click OK.

About accessing ID3 properties in MP3 files with Flash Player

Macromedia Flash Player 7 and later supports ID3 v2.4 and v2.4 tags. With this version, when you load an MP3 sound using the `attachSound()` or `loadSound()` method, the ID3 tag properties are available at the beginning of the sound data stream. The onID3 event executes when the ID3 data is initialized.

Flash Player 6 (6.0.40.0) and later supports MP3 files with ID3 v1.0 and v1.1 tags. With ID3 v1.0 and v1.1 tags, the properties are available at the end of the data stream. If a sound does not contain an ID3v1 tag, the ID3 properties are undefined. Users must have Flash Player 6 (6.0.40.0) or later for the ID3 properties to function.

For more information on using the ID3 properties, see `Sound.id3` in *Flash ActionScript Language Reference*.

Using the sound-editing controls

To define the starting point of a sound or to control the volume of the sound as it plays, you use the sound-editing controls in the Property inspector.

Flash can change the point at which a sound starts and stops playing. This is useful for making sound files smaller by removing unused sections.

To edit a sound file:

1. Add a sound to a frame (for more information, see "Adding sounds to a document" on page 203), or select a frame that already contains a sound.
2. Select Window > Properties.
3. Click the Edit button on the right side of the Property inspector.
4. Do any of the following:
   - To change the start and end points of a sound, drag the Time In and Time Out controls in the Edit Envelope.
   - To change the sound envelope, drag the envelope handles to change levels at different points in the sound. Envelope lines show the volume of the sound as it plays. To create additional envelope handles (up to eight total), click the envelope lines. To remove an envelope handle, drag it out of the window.
   - To display more or less of the sound in the window, click the Zoom In or Out buttons.
   - To switch the time units between seconds and frames, click the Seconds and Frames buttons.
5. To hear the edited sound, click the Play button.

Controlling sound playback using behaviors

You can control sound playback using sound behaviors. Behaviors are prewritten ActionScript scripts that you apply to an object, such as a button, to control a target object, such as a sound. Behaviors enable you to add the power, control, and flexibility of ActionScript coding to your document without having to create the ActionScript code yourself.

You can use the Load Sound from Library or Load Streaming MP3 File behaviors to add a sound to your document. Adding a sound using these behaviors creates an instance of the sound. The instance name is then used to control the sound.

The Play Sound, Stop Sound, and Stop All Sounds behaviors let you control sound playback. To use these behaviors, you must first load a sound with one of the Load behaviors. To play or stop a sound with a behavior, you use the Behaviors panel to apply the behavior to a triggering object, such as a button. You specify the event that triggers the behavior (such as clicking the button), select a target object (the sound to be affected by the behavior), and select settings for the behavior parameters to specify how the behavior executes.

To load a sound to a file using a behavior:

1. Select the object, such as a button, that you want to use to trigger the behavior.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button and select Sound > Load Sound from Library or Sound > Load Streaming MP3 File.
3. In the Load Sound dialog box, enter the linkage identifier for a sound from the Library, or the sound location for a streaming MP3 file. Next, enter a name for this instance of the sound, and click OK.

   For information on linkage identifiers, see "Using sounds with Sound objects" on page 205.
4. In the Behaviors panel, under Event click On Release (the default event), and select a mouse event from the menu. If you want to use the `OnRelease` event, do not change the option.

To play a sound using a behavior:

1. Select the object, such as a button, that you want to use to trigger the Play Sound behavior.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button.
3. Select Sound > Play Sound.
4. In the Play Sound dialog box, enter the instance name of the sound you want to play, and click OK.
5. In the Behaviors panel, under Event click On Release (the default event) and select a mouse event from the menu. If you want to use the `OnRelease` event, leave the option unchanged.

To stop a sound using a behavior:

1. Select the object, such as a button, that you want to use to trigger the Play Sound behavior.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button.
3. Select Sound > Stop Sound.
4. In the Stop Sound dialog box, enter the linkage identifier and the instance name of the sound you want to stop, and click OK.
5. In the Behaviors panel, under Event click On Release (the default event) and select a mouse event from the menu. If you want to use the `OnRelease` event, leave the option unchanged.

To stop all sounds using a behavior:

1. Select the object, such as a button, that you want to use to trigger the Stop All Sounds behavior.
2. In the Behaviors panel (Window > Development Panels > Behaviors), click the Add (+) button.
3. Select Sound > Stop All Sounds.
4. In the Stop All Sounds dialog box, click OK to verify that you want to stop all sounds.
5. In the Behaviors panel, under Event click On Release (the default event) and select a mouse event from the menu. If you want to use the `OnRelease` event, leave the option unchanged.

Starting and stopping sounds at keyframes

The most common sound-related task in Flash is starting and stopping sounds at keyframes to synchronize with animation.

To stop and start a sound at a keyframe:

1. Add a sound to a document. For more information, see "Adding sounds to a document" on page 203.

   To synchronize this sound with an event in the scene, select a beginning keyframe that corresponds to the keyframe of the event in the scene. You can select any of the synchronization options.
2. Create a keyframe in the sound layer's Timeline at the frame where you want the sound to end.

   A representation of the sound file appears in the Timeline.
3. Select Window > Properties and click the arrow in the lower right corner to expand the Property inspector.
4. In the Property inspector, select the same sound from the Sound pop-up menu.
5. Select Stop from the Synchronization pop-up menu.

   When you play the SWF file, the sound stops playing when it reaches the ending keyframe.
6. To play back the sound, simply move the playhead.

About the onSoundComplete event

The `onSoundComplete` event of the ActionScript Sound object lets you trigger an event in a Flash application based on completing an attached sound file. The Sound object is a built-in object that lets you control sounds in a Flash application. For more information, see "Sound class" in *Flash ActionScript Language Reference*. The `onSoundComplete` event of a Sound object is invoked automatically when the attached sound file finishes playing. If the sound is looped a specified number of times, the event is triggered when the sound finishes looping.

The Sound object has two properties that you can use with the `onSoundComplete` event. The `duration` property is a read-only property representing the duration, in milliseconds, of the sound sample attached to the sound object. The `position` property is a read-only property representing the number of milliseconds the sound has been playing in each loop.

The `onSoundComplete` event lets you manipulate sounds in a variety of powerful ways, such as the following:

- Creating a dynamic playlist or sequencer
- Creating a multimedia presentation that checks for narration completion before advancing to the next frame or scene
- Building a game that synchronizes sounds to particular events or scenes and transitions smoothly between different sounds
- Timing an image change to a sound—for example, changing an image when a sound is half over

Compressing sounds for export

You can select compression options for individual event sounds and export the sounds with those settings. You can also select compression options for individual stream sounds. However, all stream sounds in a document are exported as a single stream file, using the highest setting of all those applied to individual stream sounds. This includes stream sounds in video objects.

You select compression options for individual sounds in the Sound Properties dialog box. You can also select global compression settings for event sounds or stream sounds in the Publish Settings dialog box. These global settings are applied to individual event sounds or all stream sounds if you do not select compression settings for the sounds in the Sound Properties dialog box. For more information, see "Publishing Flash documents" on page 311.

You can also override export settings specified in the Sound Properties dialog box by selecting Override Sound Settings in the Publish Settings dialog box. This option is useful if you want to create a larger high-fidelity audio file for local use and a smaller low-fidelity version for the web. For more information, see "Setting publish options for the Flash SWF file format" on page 312.

The sampling rate and degree of compression make a significant difference in the quality and size of sounds in exported SWF files. The more you compress a sound and the lower the sampling rate, the smaller the size and the lower the quality. You should experiment to find the optimal balance between sound quality and file size.

When working with imported MP3 files, you can export the files in MP3 format using the same settings that the files had when imported.

Note: In Windows, you can also export all the sounds from a document as a WAV file using File › Export › Export Movie. For more information, see "Exporting Flash content and images" on page 345.

To set export properties for an individual sound:

1. Do one of the following:
   - Double-click the sound's icon in the Library panel.
   - Right-click (Windows) or Control-click (Macintosh) a sound file in the Library panel and select Properties from the context menu.
   - Select a sound in the Library panel and select Properties from the options menu in the upper right corner of the panel.
   - Select a sound in the Library panel and click the properties icon at the bottom of the Library panel.
2. If the sound file has been edited externally, click Update.
3. For Compression, select Default, ADPCM, MP3, Raw, or Speech. To select options for a compression format, see the following section that corresponds to the selected format:
   - "Using the ADPCM compression option" on page 210
   - "Using the MP3 compression option" on page 210
   - "Using the Raw compression option" on page 211
   - "Using the Speech compression option" on page 212

4. Set export settings.
5. Click Test to play the sound once. Click Stop if you want to stop testing the sound before it finishes playing.
6. Adjust export settings if necessary until the desired sound quality is achieved.
7. Click OK.

The Default compression option uses the global compression settings in the Publish Settings dialog box when you export your SWF file. If you select Default, no additional export settings are available.

Using the ADPCM compression option

The ADPCM compression option sets compression for 8- or 16-bit sound data. Use the ADPCM setting when you export short event sounds such as button clicks.

To use ADPCM compression:

1. In the Sound Properties dialog box, select ADPCM from the Compression menu.
2. For Preprocessing, select Convert Stereo to Mono to convert mixed stereo sounds to monaural (mono). (Mono sounds are unaffected by this option.)
3. For Sample Rate, select an option to control sound fidelity and file size. Lower rates decrease file size but can also degrade sound quality. Rate options are described in the following list:

   5 kHz is barely acceptable for speech.

   11 kHz is the lowest recommended quality for a short segment of music and is one-quarter of the standard CD rate.

   22 kHz is a popular choice for web playback and is half the standard CD rate.

   44 kHz is the standard CD audio rate.

   Note: Flash cannot increase the kHz rate of an imported sound above the rate at which it was imported.

Using the MP3 compression option

The MP3 compression option lets you export sounds with MP3 compression. Use MP3 when you are exporting longer stream sounds such as music sound tracks.

If you are exporting a file that was imported in MP3 format, you can export the file using the same settings the file had when it was imported.

To export an imported MP3 file with the same settings the file had when it was imported:

1. In the Sound Properties dialog box, select MP3 from the Compression menu.
2. Select Use Imported MP3 Quality (the default setting). Deselect this option to select other MP3 compression settings, as defined in the following procedure.

To use MP3 compression:

1. In the Sound Properties dialog box, select MP3 from the Compression menu.
2. Deselect Use Imported MP3 Quality (the default setting).
3. For Bit Rate, select an option to determine the bits per second in the exported sound file. Flash supports 8 through 160 Kbps CBR (constant bit rate). When you are exporting music, set the bit rate to 16 Kbps or higher for the best results.
4. For Preprocessing, select Convert Stereo to Mono to convert mixed stereo sounds to monaural. (Mono sounds are unaffected by this option.)

   Note: The Preprocessing option is available only if you select a bit rate of 20 Kbps or higher.

5. For Quality, select one of the following options to determine the compression speed and sound quality:

   Fast yields faster compression but lower sound quality.

   Medium yields somewhat slower compression but higher sound quality.

   Best yields the slowest compression and the highest sound quality.

Using the Raw compression option

The Raw compression option exports sounds with no sound compression.

To use raw compression:

1. In the Sound Properties dialog box, select Raw from the Compression menu.
2. For Preprocessing, select Convert Stereo to Mono to convert mixed stereo sounds to monaural. (Mono sounds are unaffected by this option.)
3. For Sample Rate, select an option to control sound fidelity and file size. Lower rates decrease file size but can also degrade sound quality. Rate options are described in the following list:

   5 kHz is barely acceptable for speech.

   11 kHz is the lowest recommended quality for a short segment of music and is one-quarter of the standard CD rate.

   22 kHz is a popular choice for web playback and is half the standard CD rate.

   44 kHz is the standard CD audio rate.

   Note: Flash cannot increase the kHz rate of an imported sound above the rate at which it was imported.

Using the Speech compression option

The Speech compression option exports sounds using a compression specially adapted to speech.

To use speech compression:

1. In the Sound Properties dialog box, select Speech from the Compression menu.
2. For Sample Rate, select an option to control sound fidelity and file size. A lower rate decreases file size but can also degrade sound quality. Select from the following options:

   5 kHz is acceptable for speech.

   11 kHz is recommended for speech.

   22 kHz is acceptable for most types of music on the web.

   44 kHz is the standard CD audio rate. However, because compression is applied, the sound is not CD quality in the SWF file.

Guidelines for exporting sound in Flash documents

Besides sampling rate and compression, there are several ways to use sound efficiently in a document and keep file size small:

- Set the in and out points to prevent silent areas from being stored in the Flash file and to reduce the size of the sound.
- Get more out of the same sounds by applying different effects for sounds (such as volume envelopes, looping, and in/out points) at different keyframes. You can get a number of sound effects using only one sound file.
- Loop short sounds for background music.
- Do not set streaming sound to loop.
- When exporting audio in embedded video clips, remember that the audio is exported using the global streaming settings selected in the Publish Settings dialog box.
- Use stream synchronization to keep the animation synchronized to your sound track when you preview your animation in the editor. If your computer is not fast enough to draw the animation frames so that they keep up with your sound track, Flash skips frames.
- When exporting QuickTime movies, use as many sounds and channels as you want without worrying about file size. The sounds are combined into a single sound track when you export as a QuickTime file. The number of sounds you use has no effect on the final file size.

Using sounds in Flash documents for mobile devices (Flash Professional only)

With Flash MX Professional 2004, you can include event sounds when authoring documents for playback on mobile devices. The general process and tools required to embed sound are described in this section. For detailed information on authoring for mobile devices, see the Content Development Kits on the Mobile and Devices Development Center at www.macromedia.com/devnet/devices.

Flash does not support sound file formats used for mobile devices (such as MIDI and others); when authoring for mobile devices, you must temporarily place a proxy sound in a supported format such as MP3, WAV, or AIFF in the Flash document. The proxy sound in the document is then linked to an external mobile device sound, such as a MIDI file. During the document publishing process, the proxy sound is replaced with the linked external sound. The SWF file generated contains the external sound and uses it for playback on a mobile device.

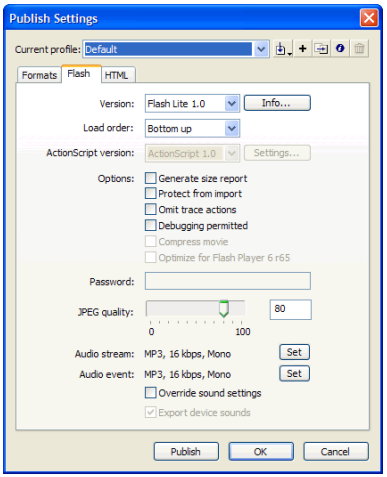
When adding sounds to Flash documents for playback on mobile devices, remember the following information:

- This feature works with event sounds only.
- The Effect, Sync, Edit, and Loop options are not supported on mobile devices.
- You must specify an external device sound file for each sound in a document.
- As with all external files, the device sound file must be available during the publishing process but is not needed by the SWF file for playback.

To add an event sound to a Flash document for playback on a mobile device:

1. Import a sound file to the library in the Flash document (File > Import > Import to Library). For information on supported file formats and importing procedures, see "Importing sounds" on page 202.
2. In the Library panel, right-click (Windows) or Control-click (Macintosh) the sound, and select Properties.
3. In the Device sound text box, enter a path or click the folder icon and browse to the location where the mobile device sound file is located. Click OK to close the Property inspector.
4. Add a button instance to the Stage from the Buttons common library (Window > Other Panels > Common Libraries > Buttons). For more information on the common libraries, see "Working with common libraries" on page 21.
5. Add the linked sound to the Hit frame of the button. For more information, see "Adding sounds to buttons" on page 204.
6. Open the Publish Settings dialog box (File > Publish Settings), and click the Flash tab.

7. Select Export Device Sounds. Flash Lite is automatically selected from the Version pop-up menu. Click OK.



The SWF file now contains the linked mobile device sound.

8. Select Control > Test Movie to test your Flash application.
9. Select Control > Disable Keyboard Shortcuts.
10. Press Tab to select the button, and then press Enter or Return to play the sound.

Note: Depending on which device you are developing for, certain restrictions might apply to how an event sound is triggered. For more information, see Mobile Articles on the Mobile and Devices Development Center at www.macromedia.com/devnet/devices.

Creating a Flash Lite sound file

Flash Lite 1.1 provides the ability to encapsulate device-specific sounds of multiple formats into a single tagged data block. This provides content developers with the ability to create a single piece of content that is compatible with multiple devices. As an example, a single Flash SFW file can contain the same sound represented in both MIDI and MFi formats. This SWF file can be played back both on a device that supports only MIDI and on a device that supports only MFi, with each device playing back the specific sound format that it natively supports.

During content creation, content developers identify the sound files in the formats that they want to bundle together. An external tool (Flash Lite Bundler.exe) is available to bundle the identified sound files into one sound data block, to be played when triggered by an event. When the appropriate event is triggered, Flash Lite 1.1 processes this bundled sound data block and plays the sound data in the specific format supported by the device.

For a complete overview of the process, see the FlashLite1.1_Authoring_Guidelines.pdf file on your Macromedia Flash application CD or on the Mobile and Devices Development Center.

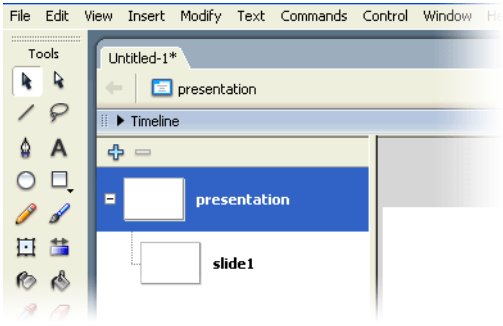
CHAPTER 12
Working with Screens (Flash Professional Only)

In Macromedia Flash MX Professional 2004, screens provide an authoring user interface with structural building blocks that make it easy for you to create complex, hierarchical Flash documents, such as slide presentations or form-based applications.

Screens provide high-level containers for creating applications. With screens, you can structure complex applications in Flash without using multiple frames and layers on the Timeline. In fact, you can create a complex application without viewing the Timeline.

When you author a screen-based document, the screens are arranged in a structured hierarchy that you create. You structure the document by nesting screens in a branching tree. You can easily preview and modify the structure of a screen-based document.

You can create screen-based documents of two types: a Flash Slide Presentation, suitable for sequential content such as a slide show or multimedia presentation, or a Flash Form Application, ideal for nonlinear, form-based applications, including Rich Internet Applications. Screen-based documents can be saved in Flash Player 6 format or later only. For an introduction to building a Flash Form Application, see the On Demand seminar, "Flash MX Professional 2004: Developing with screens," at www.macromedia.com/macromedia/events/online/ondemand/index.html.



Detail of default workspace for a new Flash Slide Presentation. Screen thumbnails appear in the Screen Outline pane on the left side of the workspace, and the Timeline is collapsed.

This chapter contains the following sections:

Understanding screen-based documents and the screen authoring environment (Flash Professional only)

The authoring environment for screen-based documents provides several ways for you to work with these documents. The next sections present information on the types of documents you can create with screens, ways to organize and navigating screens, and ways to use ActionScript, components, or Flash accessibility features with screens.

Workflow for authoring screen-based documents (Flash Professional only)

To author a screen-based document, you first create a new Slide Presentation or Form Application document. Then you add screens, configure the screens and add content, and add behaviors to create controls and transitions for the screens.

For detailed information, see the procedures described in the following sections:

- “Creating a new screen-based document (Flash Professional only)” on page 220
- “Adding screens to a document (Flash Professional only)” on page 221
- “Naming screens (Flash Professional only)” on page 222
- “Setting properties and parameters for a screen (Flash Professional only)” on page 223

- "About adding media content to screens (Flash Professional only)" on page 226
- "Selecting and moving screens (Flash Professional only)" on page 226
- "Creating controls and transitions for screens with behaviors (Flash Professional only)" on page 228

Slide presentations and form applications (Flash Professional only)

You can create screen-based documents of two types. The type of document you select determines the type of default screen in the document.

- A Flash Slide Presentation uses the slide screen as the default screen type. A slide screen has functionality designed for a sequential presentation.
- A Flash Form Application uses the form screen as the default screen type. A form screen has functionality designed for a nonlinear, form-based application.

Although each document has a default screen type, you can include both slide screens and form screens in any screen-based document. For information on slide and form screens, see "Slide screens and form screens (Flash Professional only)" on page 218.

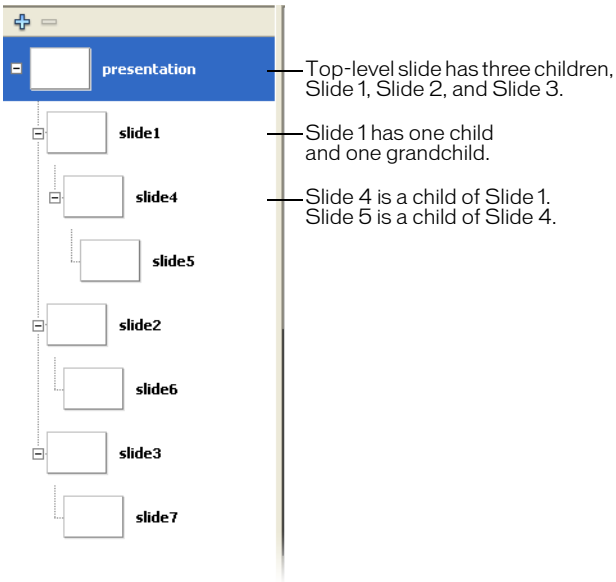
Document structure and hierarchy (Flash Professional only)

Each document has a master screen at the top level. In a Flash Slide Presentation, the top-level screen is called Presentation by default. In a Flash Form Application, the top-level screen is called Application by default.

The top-level screen is the container for everything that you add to the document, including other screens. You can place content on the top-level screen. You cannot delete or move the top-level screen.

Screens are similar to nested movie clips in some ways: Child screens inherit the behavior of their parents, and you use target paths in ActionScript to send messages from one screen to another. However, screens do not appear in the library, and you cannot create multiple instances of a screen. For information on using ActionScript with screens, see "About using ActionScript with screens (Flash Professional only)" on page 231.

You can add multiple screens to a document, and you can nest screens within other screens, in as many levels as you want. A screen that is inside another screen is the *child* of that screen. A screen that contains another screen is the *parent* of that screen. If a screen is nested several layers deep, all the screens above that screen are its *ancestors*. Screens that are at the same level are *sibling* screens. All screens nested in another screen are its *descendants*. A child screen contains all the content of its ancestor screens.



The Screen Outline pane for a Flash Slide Presentation containing screens nested three levels deep.

About using preloaders with screen-based documents

If you want to include a preloader with your screen-based document, one way to do this is to create the preloader as a separate SWF file (non-screen-based), and load the SWF file for the screen-based document from within the preloader SWF.

You cannot create a preloader within a screen-based document, because all screens in a document are located on the first frame of the root Timeline, so you cannot call or load other frames.

Slide screens and form screens (Flash Professional only)

You can create two types of screens in a document: slide screens and form screens. A Flash Slide Presentation uses the slide screen as the default screen type. A Flash Form Application uses the form screen as the default screen type. However, you can mix slide screens and form screens in any screen-based document to take advantage of the functionality of each type of screen and create more complex structure in a presentation or application.

You can set parameters for slide or form screens in the Property inspector. For more information, see "Setting parameters for a screen (Flash Professional only)" on page 225. You can also use ActionScript to control screens. For more information, see "Screen class (Flash Professional only)", "Form class (Flash Professional only)", and "Slide class (Flash Professional only)", in *Using Components*.

Slide screens let you create Flash documents with sequential content, such as a slide show. The default runtime behavior lets users navigate sequentially through slide screens, using the left and right arrow keys. Sequential screens can overlay one another so that the previous screen remains visible when the next slide is viewed. Screens can continue playing after they are hidden. Use slide screens when you want the visibility of each screen to be managed automatically.

Form screens let you create structured form-based applications, such as online registration or e-commerce forms. Form screens are simple containers that you use to structure a form-based application. By default, to create the navigation structure with form screens, you must write ActionScript. Use form screens when you want to manage the visibility of individual screens yourself.

Using the Screen Outline pane (Flash Professional only)

When you work with a screen-based document, the Screen Outline pane at the left of the Document window displays thumbnails of each screen in the current document, in a collapsible tree view. The tree represents the structural hierarchy of the document. Nested screens are indented below the screen that contains them.

When you add a screen to a document, the screen appears in the Screen Outline pane. For more information, see "Adding screens to a document (Flash Professional only)" on page 221.

You can collapse and expand the tree to hide and show nested screens. You can hide, show, and resize the Screen Outline pane.

By clicking on a screen thumbnail in the Screen Outline pane, you can display the screen on the Stage. For information on viewing screens in a document, see "Selecting and moving screens (Flash Professional only)" on page 226.

To hide or show the Screen Outline pane:

- Select Window > Screens.

To expand or collapse the tree:

- In Windows, click the Plus (+) or Minus (-) button next to a screen to show or hide the screens nested within it.
- On the Macintosh, click the triangle next to a screen to show or hide the screens nested within it.

To resize the Screen Outline pane:

- Drag the dividing line between the Screen Outline pane and the Document window.

About undoing and redoing commands with screens (Flash Professional only)

You can use the Edit > Undo and Edit > Redo menu commands to undo and redo the following actions performed on screens: adding, cutting, copying, pasting, deleting, and hiding a screen. The following actions performed on screens are recorded in the History panel: adding a screen, adding a nested screen, selecting a screen, renaming a screen, and deleting a screen. For information on the Undo and Redo commands and the History panel, see "Using the Undo, Redo, and Repeat menu commands" on page 34.

Using the screens context menu (Flash Professional only)

The screens context menu contains many commands for working with screens. You can insert screens, cut, copy and paste screens, and perform other operations with the context menu commands.

Note: Specific context menu commands are documented in sections describing those tasks. For example, to find information on the Insert Screen command, see"Adding screens to a document (Flash Professional only)" on page 221.

To view the context menu for a screen:

- Right-click (Windows) or Control-click (Macintosh) a screen thumbnail in the Screen Outline pane.

Creating a new screen-based document (Flash Professional only)

You can create a new screen-based document using one of two screen types:

- A Flash Slide Presentation uses the slide screen as the default screen type.
- A Flash Form Application uses the form screen as the default screen type.

For more information, see "Slide screens and form screens (Flash Professional only)" on page 218.

When you create a new screen-based document, it contains a top-level container screen and a single screen of the default type. Keep in mind that a screen-based document can be published only Flash Player 6 format or later, with ActionScript 2.0. You cannot save a screen-based document in any earlier Flash Player format.

You can create a new screen-based document from the Start page or from the New Document dialog box.

For information on the Start page, see "Using the Start page" in *Getting Started with Flash*. For information on the New Document dialog box, see "Creating or opening a document and setting properties" on page 12.

To create a new screen-based document from the Start page:

- Select a screen type for your document. Under Get Started, select one of the following from the Open a File options menu:

  Flash Slide Presentation creates a document with the slide screen as the default screen type.

  Flash Form Application creates a document with the form screen as the default screen type.

To create a new screen-based document from the New Document dialog box:

1. Select File > New.
2. Click the General tab and select one of the following under Type:

   Flash Slide Presentation creates a document with the slide screen as the default screen type.

   Flash Form Application creates a document with the form screen as the default screen type.

Adding screens to a document (Flash Professional only)

You can add a new screen at the same level as the currently selected screen. The new screen is a *sibling* screen of the selected screen. You can also add a nested screen one level below the currently selected screen. You can add a screen of the default screen type or select a screen type when you add a screen. You can view all screens in a document in the Screen Outline pane. For more information, see "Using the Screen Outline pane (Flash Professional only)" on page 219.

When you add screens to a document, Flash exhibits certain default behaviors:

- By default, Flash uses the screen type of the document (slide type for a Slide Presentation or form type for a Form Application) for the new screen. You can select to insert a screen of another type, using the Insert Screen Type command in the screens context menu.
- Flash inserts the first screen you add directly after the top-level screen, one level below it.
- Flash inserts a new screen after the currently selected screen, at the same level. If the document contains nested screens below the currently selected screen, the new screen is added after the nested screens, at the same level as the selected screen.
- Flash inserts a new nested screen directly after the currently selected screen, and nested one level down. If the document already contains a nested screen or screens below the currently selected screen, the new screen is inserted after all nested screens already in place, one level below the selected screen.

You can also use a template to add a new screen or a series of screens. Flash MX Professional 2004 includes screen templates in various categories.

To add a screen of the default type at the current screen level:

1. Select a screen in the Screen Outline pane.
2. Do one of the following:
   - Press Enter or Return.
   - Click the Insert Screen (+) button at the top of the Screen Outline pane.
   - Select Insert > Screen.
   - Select Insert Screen from the screens context menu.

To add a screen of a specified type at the current screen level:

1. Select a screen in the Screen Outline pane.
2. Select Insert Screen Type from the context menu and select a screen type.

To add a nested screen of the default type:

1. Select a screen in the Screen Outline pane.
2. Do one of the following:
   - Press Enter or Return.
   - Select Insert > Nested Screen.
   - Select Insert Nested Screen from the screens context menu.

To add a screen or series of screens based on a template:

1. Select a screen in the Screen Outline pane.
2. Select Insert Screen Type from the context menu and select Saved Templates.
3. Select a template category under Category, and then select a template under Templates.
4. Click OK to close the dialog box and add the template-based screen(s) to your document.

Naming screens (Flash Professional only)

By default, screens are named with their default type, in the order in which they are created: slide1, slide2, form1, form2, and so on. The creation order does not necessarily reflect the order of the screens in the Screen Outline pane. For example, you could create three sibling screens, slide1, slide2, and slide3. If you then create a nested screen directly below slide1, the nested screen is slide4.

You can rename screens, including the top-level screen. Screen names must be unique in a document. For example, you can have only one screen named *Quiz_Page* in a document.

The default screen name is used as the instance name, which is used in ActionScript to control a screen. (For more information, see "About using ActionScript with screens (Flash Professional only)" on page 231.) If you change the default screen name, the instance name is updated with the new name; likewise, if you change the instance name, the screen name is updated. The linkage identifier for the screen is also identical to the screen name, and it is updated when the screen name or instance is updated.

Instance names must conform to the following requirements:

- The name must not contain any spaces.
- The first character must be a letter, underscore (_), or dollar sign ($).
- Each subsequent character must be a letter, number, underscore, or dollar sign.
- The instance name must be unique.

You can also change the instance name in the Property inspector. For more information, see "Setting properties and parameters for a screen (Flash Professional only)" on page 223.

To rename a screen:

- Double-click the screen name in the Screen Outline pane and enter a new name.

Setting properties and parameters for a screen (Flash Professional only)

You use the Property inspector to set properties and parameters for individual screens. On the left side of the Property inspector, you can view the instance name, width, height, and *x* and *y* coordinates of a screen:

- The instance name is a unique name assigned to a screen, used when you target the screen in ActionScript. Each screen is assigned a default instance name, based on its default name in the Screen Outline pane. The instance name and default screen name are also identical to the linkage identifier for the screen. If you update the instance name, the default screen name and the linkage identifier are also updated.
- Width and height are specified in pixels. The values in the W and H fields are read-only. Width and height are determined by the screen contents. You can use the Auto Snap option to make sure the registration point stays in the same relative position when the screen width and height change. For more information, see "Specifying the ActionScript class and registration point of a screen (Flash Professional only)" on page 224.
- The *x* and *y* coordinates of a screen are specified in pixels. You can move a child screen on the Stage by changing its *x* and *y* coordinates. You can change the registration point of a screen using the registration point grid. For more information, see "Specifying the ActionScript class and registration point of a screen (Flash Professional only)" on page 224.

You can set parameters for slide and form screens, to control screen behavior during playback. For more information, see "Setting parameters for a screen (Flash Professional only)" on page 225.

To change the instance name of a screen:

1. Select a screen in the Screen Outline pane.
2. Select Window > Properties.
3. On the left side of the Property inspector, enter a name in the Instance Name text box.

   Note: If you update the instance name, the screen name in the Screen Outline pane and the linkage identifier for the screen also update.

To move a child screen on the Stage:

1. If the Hide Screen context menu option for the child screen is selected (the default setting for slide screens), deselect the option.
2. Select the screen's parent in the Screen Outline pane, and select the child screen on the Stage.
3. Select Window > Properties.
4. In the Property inspector, enter new values for the *x* and *y* coordinates, drag the child screen to another location on the Stage, or use the Align panel.

Specifying the ActionScript class and registration point of a screen (Flash Professional only)

You can specify the ActionScript class of the screen and its registration point on the Properties tab of the Property inspector:

- The ActionScript class specifies what class to which the screen belongs. The class determines what methods and properties are available for the screen. By default, slide screens are assigned to the `mx.screens.Slide` class, and form screens are assigned to the `mx.screens.Form` class. You can assign the screen to a different class.
- The registration point grid indicates the position of the screen registration point in relation to its content. By default, the registration point of a slide screen is in the center and Auto Snap is on. The registration point of a form screen is in the upper left corner and Auto Snap is off by default. You can change the registration point using the grid. You can use the Auto Snap option to keep the registration point in the same position in relation to screen contents, even when you add, remove, or reposition the screen contents.

Remember that the height and width of a screen are determined by its content. Therefore, the center of a screen cannot be the center of the Stage.

Note: If you have changed the coordinate grid setting in the Info panel in another Flash document, the coordinate grid for the screen registration point can reflect that change. To check the Info panel coordinate grid setting, open a Flash document (a non-screen-based document) or select something on the Stage that is not a screen, and select Window › Design Panels › Info. To change settings in the Info panel while working in a screen-based document, deselect all screens before you open the panel.

For more information on the Info Panel, see "Getting information about instances on the Stage" on page 67.

To change the ActionScript class of a screen:

1. Select a screen in the Screen Outline pane.
2. Select Window > Properties.
3. In the Property inspector, click the Properties tab.
4. Enter a class name in the Class Name text box. For more information on ActionScript classes, see Chapter 10, "Creating Custom Classes with ActionScript 2.0" in *Using ActionScript in Flash*.

To change the registration point of a screen:

1. Select a screen in the Screen Outline pane.
2. Select Window > Properties.
3. Click the Properties tab and click a point in the registration grid.

   Clicking a registration point automatically selects in on the Properties tab. When this option is selected, the registration point moves in relation to the screen content, but the screen itself does not move.

Setting parameters for a screen (Flash Professional only)

On the Parameters tab of the Property inspector, you can set parameters to control how the screen appears and behaves during playback. Different parameters are available for slide and form screens.

The following parameters are available only for slide screens:

- The parameter `autoKeyNav` determines whether the slide uses default keyboard handling to control navigation to the next or previous slide. When `autoKeyNav` is set to `true`, pressing the Right Arrow key or the Spacebar advances to the next slide, and pressing the Left Arrow key moves to the previous slide. When `autoKeyNav` is set to `false`, no default keyboard handling takes place. When `autoKeyNav` is set to `inherit` (the default setting), the slide inherits its `autoKeyNav` setting from its parent. If the slide's parent is also set to `inherit`, the parent's ancestors are examined until one is found with its `autoKeyNav` parameter set to `true` or `false`. If a slide is a root slide, setting `autoKeyNav` to `inherit` yields the same result as setting it to `true`.

  Note: This property can be set independently for each slide, and it affects keyboard handling when that slide has focus.

- The parameter `overlayChildren` specifies whether child screens overlay one another on the parent screen during playback. When `overlayChildren` is set to `true`, child screens overlay one another. For example, suppose you have two children, Child 1 and Child 2, which are bullet points on the parent screen. If the user clicks a Next button and displays Child 1, then clicks Next again and displays Child 2, Child 1 remains visible when Child 2 appears. When `overlayChildren` is set to `false` (the default setting), Child 1 is removed from the display when Child 2 appears. This parameter affects only the immediate children of a slide, not nested descendants.
- The parameter `playHidden` specifies whether a slide continues to play if it is hidden after being shown. When `playHidden` is set to `true` (the default setting), the slide continues to play when the slide is hidden after being shown. When `playHidden` is set to `false`, the slide stops playing if it is hidden, and resumes playing at Frame 1 if it is shown again.

There is one parameter that is available only to form screens: The parameter `visible` indicates whether a screen is visible or hidden at runtime. When `visible` is set to `true`, the screen is visible at runtime. When `visible` is set to `false`, the screen is hidden. This property does not affect the visibility of the screen in the authoring environment.

The following parameters are available for slide and form screens:

- The parameter `autoload` indicates whether the content should load automatically (`true`), or wait to load until the `Loader.load()` method is called (`false`). The default value is `true`. This parameter is inherited from the Loader component.

- The parameter `contentPath` is an absolute or relative URL indicating the file to load when the `Loader.load()` method is called. A relative path must point to the SWF file loading the content. The URL must be in the same subdomain as the URL where the Flash content currently resides. For use in Flash Player or with the Test Movie command, all SWF files must be stored in the same folder, and the filenames cannot include folder or disk drive specifications. The default value is undefined until the load starts. This parameter is inherited from the Loader component.

To specify parameter settings for a screen:

1. Select a screen in the Screen Outline pane.
2. Select Window > Properties.
3. In the Property inspector, click the Parameters tab.
4. Click the setting for a parameter, and select a setting from the pop-up menu.

About adding media content to screens (Flash Professional only)

You add media content to screens the same as you do to a Flash document that does not contain screens. You can add media content to the screen that is currently selected in the Screen Outline pane.

For general information on adding media content to a Flash document, see "About adding media content" on page 16.

Selecting and moving screens (Flash Professional only)

When you select an individual screen in the Screen Outline pane, the screen appears in the Document window. You can select multiple contiguous or discontiguous screens in the Screen Outline pane to apply modifications to several screens at once. When you select multiple screens, the contents of the first screen selected appear in the Screen Outline pane.

By default, the contents of a slide screen are not visible when you show the screen's parent in the Document window (the Hide Screen context menu option is selected). You can select to show the contents of a slide screen when its parent appears by deselecting this option. When the Hide Screen context menu option is deselected, you can select the child slide screen on the Stage. This feature affects display during authoring only, not runtime playback. (The Hide Screen context menu option is deselected for form screens by default. You can turn the option on to hide child form screens in the display during authoring.)

You can cut, copy, paste, and drag screens in the Screen Outline pane to change their position in the document, and you can remove screens from a document.

Note: The terms *child*, *parent*, and *ancestor* refer to the hierarchical relationships of nested screens. For more information, see "Document structure and hierarchy (Flash Professional only)" on page 217.

To view a screen in the Document window, do one of the following:

- Click a screen thumbnail in the Screen Outline pane to view that screen.
- With the Screen Outline pane in focus, use the keyboard keys to navigate to the screen.

- Select View > Go To and select the screen name from the submenu, or select First, Previous, Next, or Last to navigate through the screens.
- Click the Edit Screen button at the right side of the edit bar and select the screen name from the pop-up menu.

To select multiple screens in the Screen Outline pane:

- To select multiple contiguous screens, Shift-click the first and last screen you want to select.
- To select multiple discontiguous screens, Control-click (Windows) or Command-click (Macintosh) each screen.

To edit an item on a screen:

- Select the item in the Document window.

To view the contents of a child screen when the parent screen appears:

- Click Hide Screen in the child screen's context menu to turn off the Hide feature. (Hide Screen is selected for slide screens by default.)

To select a child screen on the Stage:

1. Make sure the Hide Screen context menu option is deselected. (See the previous procedure.)
2. Select the parent screen in the Screen Outline pane.
3. Click in the contents of the child screen on the Stage.

To edit an item on an ancestor screen of the current screen:

- Double-click the item in the Document window.

  The Smart Clicking feature shows the ancestor screen in the Document window and selects the item for editing.

  Note: By default, items on ancestor screens of the current screen are dimmed in the Document window.

To fully render all items on ancestor screens:

- Select View > Preview Mode > Full.

  For information on preview modes, see "Speeding up document display" on page 40.

To cut or copy a screen, do one of the following:

- Right-click (Windows) or Control-click (Macintosh) the screen, and select Cut or Copy from the context menu.
- Select Edit > Cut or Edit > Copy.

To paste a screen, do one of the following:

- After cutting or copying the screen, right-click (Windows) or Control-click (Macintosh) another screen and select Paste from the context menu. The cut or copied screen is pasted after the selected screen.

  To nest the pasted screen within the selected screen, select Paste Nested Screen from the context menu.
- After cutting or copying the screen, select Edit > Cut or Edit > Copy.

To drag a screen in the Screen Outline pane:

- Using the mouse, drag the screen to any other position in the Screen Outline pane. Release the mouse button when the screen is in the desired position. To nest a screen within another screen, drag it toward the right side of the Screen Outline pane below the intended parent.

To remove a screen:

- Do one of the following:
  - Right-click (Windows) or Control-click (Macintosh) the screen, and select Cut or Delete from the context menu.
  - Select the screen, and click the Delete Screen (-) button at the top of the Screen Outline pane.
  - Press Backspace (Windows) or Delete (Macintosh).

Creating controls and transitions for screens with behaviors (Flash Professional only)

You can create controls and transitions for screens using behaviors. Controls enable the flow between screens—for example, you can go to another screen, hide a screen, or show a screen. Transitions create visual animations that play as the Flash document display changes from one screen to another.

Behaviors are built-in ActionScript scripts that you add to an object, such as a screen, to control that object. Behaviors lets you add the power, control, and flexibility of ActionScript coding to your document without having to create the ActionScript code yourself. Behaviors are available for a variety of objects in Flash, including movie clips, text fields, and video and sound files.

Adding controls to screens using behaviors (Flash Professional only)

To add a control to a screen using a behavior, you attach the behavior to a trigger—a button, movie clip, or screen—and target the screen that you want to affect by the behavior. You can select the event that triggers the behavior.

You can add the following behaviors to control slide screens: Go to First Slide, Go to Last Slide, Go to Next Slide, Go to Previous Slide, and Go to Slide (specify slide name).

Note: Go to Next Slide and Go to Previous Slide move to screens on the same level, not to parents or children. For an explanation of parents and children, see "Document structure and hierarchy (Flash Professional only)" on page 217.

You can add the following behaviors to control slide or form screens: Show a Specified Screen (if the screen has previously been hidden) or Hide a Specified Screen (if the screen has previously been shown).

To add a control behavior:

1. Select the button, movie clip, or screen to trigger the behavior.
2. In the Behaviors panel, click the Add (+) button.
3. Select Screen, and select the desired control behavior from the submenu.

4. If the behavior requires that you select a target screen, the Select Screen dialog box appears. Select the target screen in the tree control. Click Relative to use a relative target path, or Absolute to use an absolute target path, and click OK. (For information on target paths, see "Using absolute and relative target paths" on page 23.)

   Note: Some behaviors select a target screen by default; for example, the Go to First Slide screen automatically targets the first screen. These behaviors do not show the Select Screen dialog box.

5. In the Event column, click in the row for the new behavior and select an event from the list. This specifies the event that triggers the behavior—for example, a user clicking a button, a movie clip loading, or a screen receiving focus. The list of available events depends on the type of object you use to trigger the behavior.

Adding transitions to screens using behaviors (Flash Professional only)

Screen transition behaviors let you add animated transitions between screens, fade a screen in or out, rotate a screen as it appears or disappears, have a screen fly in from the edge of a document, and create other effects. To add a transition using a behavior, you attach the behavior directly to a screen.

You can select the direction of a transition: In, which plays the animation as the screen first appears in the document; or Out, which plays the animation as the screen disappears from the document. You can also specify the duration in seconds.

Easing options let you modify the transition to achieve different effects. For example, the Bounce easing option makes the screen appear to bounce as the transition completes.

Some transitions have additional parameters that you can modify. Parameters appear in the Transitions dialog box when you select the transition.

Follow these guidelines when adding transitions:

- For most situations, the In option is recommended.
- Use the In option when applying a transition that uses the `on(reveal)` event.
- Use the Out option when applying a transition that uses the `on(hide)` event.
- Do not add an Out transition immediately before an In transition in a presentation.
- To attach the same transition to all children of a given slide, attach the single transition to the `on(revealChild)` or `on(hideChild)` event of the parent, rather than duplicating the transition on all child slides.

To add a transition behavior:

1. Select the screen to which you want to apply the behavior.
2. In the Behaviors panel, click the Add (+) button.
3. Select Screen > Transition from the submenu.
4. In the Transition dialog box, select a transition from the scroll list.

   An animated preview of the transition plays in the preview window, and a brief description of the transition appears in the description field. The animation changes to reflect options that you select for the transition in the following steps.

5. For Direction, select In to play the transition as the screen appears in the document, and Out to play the transition as the screen disappears from the document.
6. For Duration, enter a time in seconds.
7. For Easing, select an option to define the transition style.
8. If the transition has additional parameters, select options or enter values for those parameters in the fields provided.
9. Click OK.
10. In the Behaviors panel, go to the Event column and click in the row for the new behavior, and select an event from the list. This action specifies the event that triggers the behavior—for example, the mouse pointer moving over the screen.

Using Find and Replace with screens (Flash Professional only)

You can use the Find and Replace feature to find and replace a specified element in a Flash document that uses screens. You can search for a text string, font, color, symbol, sound file, video file, or imported bitmap file.

You can search for elements in the entire document or in the current screen.

To use Find and Replace with a document containing screens:

1. Select Edit > Find and Replace.
2. Do one of the following:

- To search the entire document, select Current Document from the Search In pop-up menu.
- To search a screen, click in the Screen Outline pane, and select Current Screen from the Search In pop-up menu.

  For instructions on searching for text, fonts, colors, and so on, see "Using Find and Replace" on page 30.

About using the Movie Explorer with screens (Flash Professional only)

You can use the Movie Explorer to view and organize the contents of a document containing screens. The Movie Explorer handles documents that contain screens much as it handles those that do not contain screens, with the following exceptions:

- The Movie Explorer shows the contents of the current screen (the screen selected in the Screens Outline pane) only.
- You cannot view scenes in the Movie Explorer because a document with screens cannot contain scenes.

For more information, see "Using the Movie Explorer" on page 28.

About using Timelines with screens (Flash Professional only)

Each screen has its own Timeline. The Timeline is collapsed by default. You must expand it to work with frames or layers.

You cannot view or modify the main Timeline of a screen-based document.

You can add frames, keyframes, and layers, and manipulate content on a screen's Timeline.

In the Timeline, nested screens work much as nested movie clips do, with some exceptions. For more information, see "How screens interact with ActionScript (Flash Professional only)" on page 232.

About using ActionScript with screens (Flash Professional only)

You can use ActionScript to control screens in a document. You can insert, remove, rename, or change the order of screens, and perform other operations.

ActionScript uses the screen instance name, class name, and registration point when controlling screens. For more information, see "Screen instance names, class names, and registration points (Flash Professional only)" on page 231. ActionScript also uses the screen parameters. For more information, see "Setting parameters for a screen (Flash Professional only)" on page 225.

Screens and movie clips interact with ActionScript in similar ways, but with some important differences. For more information, see "How screens interact with ActionScript (Flash Professional only)" on page 232.

For more information, see "Screen class (Flash Professional only)", "Form class (Flash Professional only)"" and "Slide class (Flash Professional only)", in *Using Components*.

Screen instance names, class names, and registration points (Flash Professional only)

The screen name automatically generates the instance name and class name of the screen. You need these identifying labels when you manipulate screens with ActionScript in various ways. You can change a screen's registration point to adjust how the screen behaves. You can work with these features in various ways, as described in the following list:

- The instance name is a unique name assigned to a screen, used when you target the screen in ActionScript. You can change the instance name in the Property inspector. The instance name is identical to the screen name in the Screen Outline pane and the linkage identifier for the screen. If you update the instance name, the screen name and the linkage identifier also update. For more information, see "Setting properties and parameters for a screen (Flash Professional only)" on page 223.

  Note: Symbol instances, including movie clips, buttons, and graphics, also have instance names. For more information on symbol instances, see Chapter 3, "Using Symbols, Instances, and Library Assets," on page 53.

- The class name identifies the ActionScript class to which the screen is assigned. By default, a slide screen is assigned to the `mx.screens.Slide` class, and a form screen is assigned to the `mx.screens.Form` class. You can assign the screen to a different class to modify the methods and properties that are available for the screen. For more information on ActionScript classes, see Chapter 10, "Creating Custom Classes with ActionScript 2.0" in *Using ActionScript in Flash*.
- The Property inspector indicates the registration point in the *x* and *y* coordinate fields and in the registration point grid. For more information, see "Setting properties and parameters for a screen (Flash Professional only)" on page 223. You might want to move the registration point for greater control in manipulating screen content. For example, if you want to create a spinning shape in the center of a screen, you can reposition the screen registration point at the center of the screen and rotate the screen around its registration point.

How screens interact with ActionScript (Flash Professional only)

Screens are similar to nested movie clips in the way that they interact with ActionScript. (For more information, see "Nested movie clips" on page 22.) However, there are some differences.

Remember the following guidelines when you use ActionScript with screens:

- When you select a screen in the Screen Outline pane and add ActionScript, the script is added directly to the screen as an object action (much as ActionScript is added directly to a movie clip). It's usually best to use object actions for simple code (such as creating navigation between screens) and external ActionScript files for more complex code.
- For best results, organize the document structure and finalize screen names before adding ActionScript. If you rename a screen, the instance name is automatically changed, and you must update the instance names in any ActionScript code you have written.
- If you want to add a frame action to the Timeline for a screen, you must select the screen, expand the Timeline (collapsed by default), and select the first frame in the Timeline. However, it's usually best to use an external ActionScript file, rather than a frame action, for complex code on a screen.
- You cannot view or manipulate the main Timeline for a screen-based document. However, you can target the main Timeline using `_root` in a target path.
- Each screen is automatically associated with ActionScript, based on its class. (For more information, see "Slide screens and form screens (Flash Professional only)" on page 218.) You can change the class to which that screen is assigned, and you can set some parameters for a screen in the Property inspector. For more information, see "Setting properties and parameters for a screen (Flash Professional only)" on page 223.
- Use the Screen class, Slide class, and Form class to control screens with ActionScript.
- Use components whenever possible to create interactivity. Put no more than 125 total component instances in a single FLA file.
- To create navigation between slides, use `rootSlide`. For example, to get the current slide, use `rootSlide.currentSlide`.
- Do not try to do slide navigation inside of `on(reveal)` or `on(hide)` handlers.
- Do not add an `on(keydown)` or `on(keyup)` event to ActionScript code controlling a screen.

For more information on controlling screens with ActionScript, see “Screen class (Flash Professional only)”, “Form class (Flash Professional only)”, and “Slide class (Flash Professional only)”, in *Using Components*.

For information on the Object class and the `onclipEvent()` event handler, see “Object class” and `onClipEvent()` in *Flash ActionScript Language Reference*.

About using components with screens (Flash Professional only)

You can use components with screens to create complex, structured applications in Flash. Components are especially useful with forms, to create structured applications that show data and enable nonlinear user interactivity. For example, you can use forms to populate a container component.

When you use components with screens, you can use the Focus Manager to create custom navigation between components. The Focus Manager specifies the order in which components receive focus when a user presses the Tab key to navigate in an application. For example, you can customize a form application so that a user can press Tab to navigate fields and press Return (Macintosh) or Enter (Windows) to submit the form.

For information on the Focus Manager, see “Creating custom focus navigation” and “FocusManager class” in *Using Components*.

You can also create a tab order using the Accessibility panel. For more information, see “Viewing and creating tab order and reading order” on page 366.

Accessibility in the Flash screens authoring environment (Flash Professional only)

Accessibility support is available for screen-based documents in the Flash authoring environment. Using keyboard shortcuts rather than the mouse, users can navigate a document and use interface elements, including screens, panels, the Property inspector, dialog boxes, the Stage, and objects on the Stage.

Accessibility support for screen-based documents is similar to support for other documents, with one exception: when keyboard shortcuts are used to navigate panels (Control+Alt+Tab in Windows or Command+Option+Tab on the Macintosh), the Screen Outline pane receives focus the first time the keyboard shortcut is used. (For other documents, the Timeline receives focus first.)

To cycle through individual screens in the Screen Outline pane, you use the arrow keys.

The Screen Outline pane receives focus only the first time you cycle through the panels. That is, if you come to the last panel and press the keyboard shortcut again, the Screen Outline pane is skipped, and the next panel receives focus.

For complete information on accessibility in the Flash authoring environment, see Chapter 17, “Creating Accessible Content,” on page 355.

CHAPTER 13
Creating Multilanguage Text

As more applications are distributed to worldwide audiences, it is becoming common to author applications that can appear in multiple languages. Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 provide several new features that greatly enhance the work flow for authoring multiple language Unicode-based applications. You can include multilanguage text in your document in the following ways:

- The new Strings panel lets localizers edit strings in a central location inside Flash or in external XML files with their preferred software or translation memory. For more information, see "Authoring multilanguage text with the Strings panel" on page 240.
- You can select which character sets you want to embed in your applications, which limits the number of character glyphs in your published SWF file and reduces its size. For more information, see "Using embedded fonts" on page 238.
- You can use a Western-style keyboard to create text on the Stage in Chinese, Japanese, and Korean. For more information, see "Using a Western keyboard to enter Asian characters on the Stage" on page 248.
- If you have Unicode fonts installed on your system, you can enter text directly into a text field. Because the fonts are not embedded, your users must also have Unicode fonts. For more information, see "Creating documents with multilanguage text without using the Strings panel" on page 248.

Other, less common, methods of including multilingual text in your movie include the following:

- You can include an external text file in a dynamic or input text field, using the `#include` action. For more information, see "Creating documents with multilanguage text using the #include action" on page 249.
- You can load external text or XML files into a Flash application at runtime using the `loadVariables` action, the `getURL` action, the `LoadVars` object, or the `XML` object. For more information, see "Using ActionScript to load external files" on page 248.
- You can enter Unicode escape characters in the string value for a dynamic or input text field variable. For more information, see "Creating documents with multilanguage text using text variables" on page 250.

For Unicode-encoded text to appear correctly, users must have access to fonts containing the glyphs (characters) used in that text. For more information, see "Using external text or XML files that are not Unicode encoded" on page 251.

This chapter contains the following sections:

Selecting an encoding language

All text in a computer is encoded as a series of bytes. Many different forms of encoding (and therefore, different bytes) represent text. Different kinds of operating systems use different kinds of encoding for text. For example, Western Windows operating systems usually use CP1252 encoding; Western Macintosh operating systems usually use MacRoman encoding; Japanese Windows and Macintosh systems usually use Unicode encoding.

Unicode can encode most languages and characters used throughout the world. The other forms of text encoding used by computers are subsets of the Unicode format, tailored to specific regions of the world. Some of these forms are compatible in some ranges and incompatible in other ranges, so using the correct encoding is critical.

Unicode comes in several forms. Flash Player versions 6 and 7 support text or external files in the 8-bit Unicode format UTF-8, and in the 16-bit Unicode formats UTF-16 BE (Big Endian) and UTF-16 LE (Little Endian). For more information, see "Text encoding in Flash Player 7" on page 237.

Unicode and Macromedia Flash Player

Macromedia Flash Player 6 and later supports Unicode text encoding. Any user with Flash Player 6 or later can view multilanguage text, regardless of the language used by the operating system running the player, if they have the correct fonts installed.

Flash Player 6 and later assumes that all external text files associated with a Flash Player application are Unicode encoded, unless you tell the player otherwise. If you use external text files that are not Unicode encoded, you can set the `system.useCodepage` property to `true` to tell Flash Player to use the traditional code page of the operating system running the player. For more information, see "Using external text or XML files that are not Unicode encoded" on page 251.

For Flash applications in Macromedia Flash Player 5 or earlier that are authored in Flash MX or earlier, Flash Player 6 and earlier versions display the text using the traditional code page of the operating system running the player.

For background information on Unicode, see www.Unicode.org.

Text encoding in Flash Player 7

By default, Flash Player 7 assumes that all text it encounters is Unicode encoded. If your document loads external text or XML files, the text in these files should be UTF-8 encoded. You can create these files using the Strings panel or in a text or HTML editor, such as Macromedia Dreamweaver MX 2004, that can save the files in Unicode format.

Flash Player 7 supports the 8-bit Unicode format UTF-8, and the 16-bit Unicode formats UTF-16 BE (Big Endian) and UTF-16 LE (Little Endian). For more information, see "Unicode encoding formats supported by Flash Player" on page 237.

Unicode encoding formats supported by Flash Player

When reading text data in Flash, Flash Player looks at the first two bytes in the file to detect a byte order mark (BOM), a standard formatting convention used to identify the Unicode encoding format. If no BOM is detected, the text encoding is interpreted as UTF-8 (an 8-bit encoding format). It is recommended that you use UTF-8 encoding in your applications.

If Flash Player detects either of the following BOMs, the text encoding format is interpreted as follows:

- If the first byte of the file is OxFE and the second is OxFF, the encoding is interpreted as UTF-16 BE (Big Endian). This is used for Macintosh operating systems.
- If the first byte of the file is OxFF and the second is OxFE, the encoding is interpreted as UTF-16 LE (Little Endian). This is used for Windows operating systems.

Most text editors that can save files in UTF-16BE or LE automatically add the BOMs to the files.

Note: If you set the `system.useCodepage` property to `true`, the text is interpreted using the traditional code page of the operating system that is running the player; it is not interpreted as Unicode. For more information, see "Using external text or XML files that are not Unicode encoded" on page 251.

About encoding in external XML files

You cannot change the encoding of an XML file by changing the encoding tag. Flash Player identifies the encoding of an external XML file using the same rules as for all external files: If no BOM is encountered at the beginning of the file, the file is assumed to be in UTF-8 encoding. If a BOM is encountered, the file is interpreted as UTF-16BE or LE. For more information, see "Unicode encoding formats supported by Flash Player" on page 237.

Fonts for Unicode-encoded text

When you use external files that are Unicode encoded, your users must have access to fonts containing all the glyphs used in your text files. By default, Flash MX 2004 stores the names of fonts used in dynamic or input text files. During SWF file playback, Flash Player 7 (and earlier versions) looks for those fonts on the operating system running the player.

If the text in a SWF file contains glyphs that are not supported by the specified font, Flash Player 7 attempts to locate a font on the user's system that supports those glyphs. It is not always possible for the player to locate an appropriate font. The behavior of this function depends on the fonts available on the user's system as well as on the operating system running Flash Player.

Using embedded fonts

You can embed fonts for dynamic or input text fields. However, some fonts, particularly those used for Asian languages, can add significantly to the SWF file size when embedded. With Flash MX 2004 and Flash MX Professional 2004, you can select ranges of fonts you want to embed.

To select and embed a range of fonts:

1. On the Stage, select a text field, and then show the Property inspector (Window > Properties).
2. Click the Character button to show the Character Options dialog box.
3. Select one of the following options:

   No Characters Select this option if you do not want to embed any characters, but rather use the font specified during authoring or provide appropriate font substitution when using device fonts.

   Specify Ranges Select this option to select a range of characters to embed into the SWF file. By selecting only the characters you want to embed, you can create a smaller, more efficient SWF file.
4. If you have selected Specify ranges, select the ranges of font sets you want to embed by doing the following:
   - Click on a font set in the pop-up menu.
   - Select multiple ranges by Shift-clicking the first and last fonts of a continguous range of fonts, or by Control-clicking (Windows) or Command-clicking (Macintosh) to select noncontiguous fonts.

   The size of each font group appears in parentheses next to the font name. As you select multiple font sets, the panel shows the total number of glyphs you selected.

   Note: For example, to embed Chinese and Western characters, you would need to select Chinese and Western font sets. Select only the font sets you want to embed, however, so you do not exceed the internal maximum number of glyphs for the authoring tool (approximately 30,000). If you select more than the maximum, a warning dialog box appears.
5. Click OK. If you exceed the internal maximum number of glyphs for the authoring tool a warning dialog box appears.

   Note: Flash does not perform error-checking to confirm that glyphs actually exist in the font for the selected character set. During the actual publish or export procedure, only glyphs that are present in the font are embedded in the SWF file.

To embed font sets from text on the Stage:

1. Select the text on the Stage.
2. In the Property inspector (Window > Properties), click the Character button to show the Character Options dialog box.
3. If necessary, select Specify Ranges.
4. Click the AutoFill button.
5. Click OK.

   The font glyphs for the selected fonts are embedded.

XML font embedding table

The list of selected fonts is stored and maintained as an external XML file and resides in the user configuration folder. It is named Unicode_Table.xml and contains the one-to-many relationship between a particular language and all the necessary Unicode glyph ranges as shown in the following Korean examples.

The font set groupings are based upon the Unicode Blocks as defined by the Unicode Consortium. They are organized in Flash in a manner that makes selection quick and easy. To provide a simpler workflow, when you select a particular language, all related glyph ranges are embedded even if they are scattered into disjointed groupings.

For example, if you select Korean, the following Unicode character ranges are embedded.

3131-318E Hangul symbols

3200-321C Hangul specials

3260-327B Hangul specials

327F-327F Korean symbol

AC00-D7A3 Hangul symbols

If you select Korean + CJK, a larger font set is embedded:

3131-318E Hangul symbols

3200-321C Hangul specials

3260-327B Hangul specials

327F-327F Korean symbol

4E00-9FA5 CJK symbols

AC00-D7A3 Hangul symbols

F900-FA2D CJK symbols

Font set selections

The following table gives more details about each font set selection:

Range	Description
Uppercase [A-Z]	Basic Latin uppercase glyphs
Lowercase [a-z]	Basic Latin lowercase glyphs
Numerals [0-9	Basic Latin numeral glyphs
Punctuation [!@#%...]	Basic Latin punctuation
Basic Latin	Basic Latin glyphs within the Unicode range 0x0021 to 0x007E
Japanese Kana	Hiragana and Katakana glyphs (including half-width forms)
Japanese Kanji - Level 1	Japanese Kanji characters
Japanese (All)	Japanese Kana and Kanji (including punctuation and special characters)

Range	Description
Basic Hangul	Most commonly used Korean characters, Roman characters, punctuations, and special characters/symbols
Hangul (All)	11,720 Korean characters (sorted by Hangul syllables), Roman characters, punctuations, and special characters/symbols)
Traditional Chinese - Level 1	5000 most commonly used Traditional Chinese characters used in Taiwan
Traditional Chinese (All)	All Traditional Chinese characters used in Taiwan and Hong Kong, and punctuations
Simplified Chinese - Level 1	6000 most commonly used Simplified Chinese characters used in mainland of China and punctuations
Chinese (All)	All Traditional and Simplified Chinese characters and punctuations
Thai	All Thai glyphs
Devanagari	All Devanagari glyphs
Latin I	Latin-1 Supplement range 0x00A1 to 0x00FF (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)
Latin Extended A	Latin Extended-A range 0x0100 to 0x01FF (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)
Latin Extended B	Latin Extended-B range 0x0180 to 0x024F (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)
Latin Extended Add'l	Latin Extended Additional range 0x1E00 to 0x1EFF (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)
Greek	Greek and Coptic, plus Greek Extended (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)
Cyrillic	Cyrillic (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)
Armenian	Armenian plus ligatures
Arabic	Arabic plus Presentation Forms-A and Presentation Forms-B
Hebrew	Hebrew plus Presentation Forms (including punctuation, superscripts and subscripts, currency symbols, and letterlike symbols)

Authoring multilanguage text with the Strings panel

The new Strings panel provides a simplified workflow for authoring multilanguage text. The general workflow steps are described in the following list:

- Author a FLA file in one language. Any text that you want to enter in another language must be in a dynamic or input text field.
- In the Strings panel Settings dialog box, select the languages you want to include and select one of them as the default language.

- After you select a language, a column for the language is added to the Strings panel. When you save, test, or publish the application, a folder with an XML file is created for each language. For more information, see "Selecting languages for translation" on page 241.
- In the Strings panel, encode each text string with an ID. For more information, see "Adding strings to the Strings panel" on page 242.
- Publish the application.
- A folder is created for each language you select, and within each language folder is an XML file for that language. For more information, see "Publishing and deploying multilanguage text" on page 244.
- Send the published FLA file and XML folders and files to your translators. You can author in your native language, and let them make the translation. They can use translation software directly in the XML files or in the FLA file. "Translating text in the Strings panel or an XML file" on page 246.
- When you receive the translations from your translators, import the translated XML files back into the FLA file. For more information, see "Importing an XML file into the Strings panel" on page 247.

For a sample of a multilanguage document, see "Developing multilingual content" on page 422.

Selecting languages for translation

You can select as many as 100 languages that can appear on the Stage and in the Strings panel for translation. Each language you select becomes a column in the Strings panel. You can change the Stage language to show the text on the Stage in any of the languages you selected. The selected language appears when you publish or test the file.

When selecting languages, you can use any of the languages provided in the pop-up menu as well as any other Unicode-supported language.

To select a language:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. Click the Settings button to show the Settings dialog box.
3. Add a language by doing one of the following:
   - In the Languages pop-up menu, highlight a language you want to select, and click the Add button.
   - If the language does not appear in the pop-up menu, in the blank field below the Select languages pop-up menu, type a language code and optional country code, in the format xx_XX. Then click the Add button. The first part, xx, is the language code from ISO 639-1, and XX is the optional uppercase two-letter country code from ISO 3166-1.

   After you click the Add button, the language appears in the Available Languages field.
4. Repeat step 3 until you have added all the languages you want.
5. In the Default language field, select a default language. This language appears on systems that do not have one of the available languages you selected.

6. If you want to load an XML file for the languages from a different URL at runtime, type the URL in the URL text box.

   Note: The XML file generated by Flash is stored in the folder indicated by the SWF publish path. For more information, see "Publishing Flash documents" on page 311. If no SWF publish path is selected, the XML file is stored in a folder for the language within the folder for the FLA file.

7. Click OK.

   A column for each selected language appears in the Strings panel. The columns appear in alphabetical order.

8. Save the FLA file. When you save the FLA file, a folder for each language you selected is created in the same folder indicated in the SWF publish path. For more information, see "Publishing Flash documents" on page 311. If no SWF publish path has been selected, it is created in the folder the FLA file resides in. Within each language file an XML file is created that is used to load translated text.

To remove a language:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. Click the Settings button to show the Settings dialog box.
3. In the Available Languages field, highlight a language you want to remove, and click the Remove button.

   The language no longer appears in the Available Languages field.

4. Repeat step 3 until you have removed all the languages you want.
5. When you are done removing languages, click OK.

   The column for each removed language no longer appears in the Strings panel.

Adding strings to the Strings panel

There are several ways to assign text strings to the Strings panel: you can assign a string ID to a dynamic or input text field, add a string to the Strings panel without assigning it to a text field, or assign an existing string ID to an existing dynamic or input text field. For information about creating dynamic and input text fields, see "Creating text" on page 108.

To assign a string ID to a text field:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. Select the Text tool from the toolbar. On the Stage, create an input or dynamic text field.
3. While the text field is selected, enter a unique ID in the ID field in the Strings panel.

   Note: If a static text box is selected on the Stage, the Stage text selection section on the Strings panel displays the message "Current text cannot have an ID associated with it." If a nontext item is selected or multiple items are selected, it shows the message "Current selection cannot have an ID associated with it."

4. In the Strings panel, type the string in the String text box.
5. Click Apply to add the string to the Strings panel.

   Note: You can also use the Enter key to apply the ID to the text field.

To add a string ID to the Strings panel without assigning it to a text field:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. Type in a new string ID and new string in the Strings panel, and click the Apply button.

   Note: You can also use the Enter key to apply the ID to the text field.

3. When you are ready to assign the new string to a text field, perform the steps in the following procedure.

To assign an existing ID to a text field:

1. Select the Text tool from the toolbar. On the Stage, create an input or dynamic text field.
2. Type the name of an existing ID in the ID section of the Strings panel.
3. Click Apply.

   The String text field on the Stage appears with the text string assigned to the ID.

   Note: You can also use the Enter key to apply the ID to the text field.

Changing the language displayed on the Stage

You can change the language that appears on the Stage to any of the available languages you selected. For more information, see "Selecting languages for translation" on page 241.

To display the text on the Stage in another language:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. In the Stage Language pop-up menu, select the language you want to use for the Stage language. This must be a language you added as an available language.

   After you change the Stage language, any new text you type on the Stage appears in that language. If you have previously entered text strings for the language in the Strings panel, any text on the Stage appears in the selected language. If not, the text fields already on the Stage are blank.

About editing text in the Strings panel

After you have entered text in the Strings panel, you can edit the text in the following ways:

- You can edit the text directly in the Strings panel cells.
- You can edit the text on the Stage in the language selected as the Stage language, using language-editing features such as find and replace (see "Finding and replacing text" on page 30) and spell checking (see "Using the Check Spelling feature" on page 118). Text that is changed using these features is changed on the Stage and in the Strings panel.
- You can edit the XML file directly. For more information, see "Translating text in the Strings panel or an XML file" on page 246.

Publishing and deploying multilanguage text

When you save, publish, or test the FLA file, a folder with an XML file is created for each available language you selected in the Strings panel. The default location for the XML folders and files is the same folder indicated as the SWF publish path. For more information, see ""Publishing Flash documents" on page 311. If no SWF publish path has been selected, the XML folder and files are saved in the folder in which the FLA file is located. For example, if you have a file named Test in the mystuff folder, and you have selected English (en), German (de), and Spanish (es) as available languages, and you have not selected a SWF publish path, when you save the FLA file, the following folder structure is created:

```
\mystuff\Test.fla
\mystuff\de\Test_de.xml
\mystuff\en\Test_en.xml
\mystuff\es\Test_es.xml
```

When you deploy a SWF file, you also need to deploy the associated XML files with the string translations in the web server. The first frame that contains text cannot appear until the entire XML file is downloaded.

Automatic language detection and the default language

You can change the default language to any language that you have selected as an available language. When automatic language detection is turned on, and the SWF file is viewed on the language operating system platform, the default language appears on any system that is set to a default language other than those languages you selected. For example, if you set your default language to English, and if you selected JP, EN, and FR as available languages, users who have their system language set to Japanese, English, or French automatically see text strings in their language. However, users who have their system language set to Swedish, which is not one of the selected languages, automatically see text strings in the default language you selected—in this case, English.

When you publish a Flash application (select File > Publish), however, the default language setting specified in the Strings panel Settings dialog box is published in the resulting SWF file. For example, if the default language set in the Strings panel Settings dialog box is French, but the operating system and Flash are in English, the published SWF file is in the default language, which, in this case, is French.

Note: When you test a Flash application (select Control › Test Movie), the Stage language setting specified in the Strings panel is published in the resulting SWF file. For example, if the Stage language set in the Strings panel Settings dialogue box is English, and the default language is French, the test SWF file is in the Stage language, in this case English.

To select the default language and automatic language detection:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. Click the Settings button to show the Settings dialog box.
3. In the Default language pop-up menu, select the language you want to use for the default language. This must be a language you have added as an available language.

4. If you want to enable automatic language detection, make sure that Insert ActionScript for Automatic Language Detection is selected.
5. Click OK.

XML file format

Exported XML is in UTF-8 format and follows the XML Localization Interchange File Format (XLIFF)1.0 standard. It defines a specification for an extensible localization interchange format that lets any software provider produce a single interchange format that can be delivered to, and understood by, any localization service provider. For more information about XLIFF, see www.oasis-open.org/committees/xliff/.

XLIFF examples

If any of the following characters are entered in the Strings panel, they are replaced by the appropriate entity reference when written to XML files:

Character	Replaced by
&	&amp;
'	&apos;
"	&quot;
<	&lt;
>	&gt;

Exported XML file sample

The following samples show what an XML file generated by the Strings panel looks like in the source language—in this example, English—and in another language—in this example, French:

English source version sample

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE xliff PUBLIC "-//XLIFF//DTD XLIFF//EN"
"http://www.oasis-open.org/committees/xliff/documents/xliff.dtd" >
<xliff version="1.0" xml:lang="en">
<file datatype="plaintext" original="MultiLingualContent.fla" source-
  language="EN">
    <header></header>
    <body>
      <trans-unit id="001" resname="IDS_GREETINGS">
        <source>welcome to our web site!</source>
      </trans-unit>
      <trans-unit id="002" resname="IDS_MAILING LIST">
        <source>Would you like to be on our mailing list?</source>
      </trans-unit>
      <trans-unit id="003" resname="IDS_SEE YOU">
        <source>see you soon!</source>
      </trans-unit>
      <trans-unit id="004" resname="IDS_TEST">
```

```
        <source></source>
      </trans-unit>
    </body>
  </file>
</xliff>
```

French version sample

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE xliff PUBLIC "-//XLIFF//DTD XLIFF//EN"
"http://www.oasis-open.org/committees/xliff/documents/xliff.dtd" >
<xliff version="1.0" xml:lang="fr">
<file datatype="plaintext" original="MultiLingualContent.fla" source-
  language="EN">
    <header></header>
    <body>
      <trans-unit id="001" resname="IDS_GREETINGS">
        <source>Bienvenue sur notre site web!</source>
      </trans-unit>
      <trans-unit id="002" resname="IDS_MAILING LIST">
        <source>Voudriez-vous être sur notre liste de diffusion?</source>
      </trans-unit>
      <trans-unit id="003" resname="IDS_SEE YOU">
        <source>A bientôt!</source>
      </trans-unit>
      <trans-unit id="004" resname="IDS_TEST">
        <source></source>
      </trans-unit>
    </body>
  </file>
</xliff>
```

Translating text in the Strings panel or an XML file

After you finish authoring your document, have assigned IDs to all the text in the Strings panel and selected all the languages into which you want to translate the document, you can send it to translators. When sending files to translators, you need to include not only the FLA file but also the folders for the XML files and the XML file for each language.

Translators can either work directly in the language columns in the Strings panel or work in the XML files for each language to translate the FLA file to selected languages. If you translate directly in the XML file, you must either import the XML file to the Strings panel or save it in the default directory for that language. For more information, see “Importing an XML file into the Strings panel” on page 247.

To translate text in the Strings panel:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. For each language to be translated, select the appropriate language column, then type the translated text for that language to be associated with each string ID.
3. To show the text on the Stage in the language you selected, select the language in the Stage Language field.

4. When you are finished, save, publish, or test the file.

   All XML files for all languages are overwritten with the information in the Strings panel.

   Note: If you want to preserve the translation in an XML file, save it in a different folder.

To translate text in an XML file:

1. Using an XML file editor or translating software, open the folder for the desired language, then the XML file for that language. The XML file is populated with the IDs for each text string.
2. Enter the text string for the language next to the ID. For more information, see "English source version sample" on page 245 and "French version sample" on page 246.
3. If necessary, import the translated XML file into the Strings panel. For more information, see the following section.

Importing an XML file into the Strings panel

After you modify an XML file, if you place it in the folder specified in the Strings panel for that language, the XML file is loaded into the FLA file when it opens.

You can also import an XML file into the Strings panel from another location. After you import it, when you save, test, or publish the file, the XML file in the folder specified for that language is overwritten. You cannot import an XML file for a language unless it has already been selected as an available language in the Strings panel. You can also add a language and import an XML file with the translation for that language.

To import an XML file into the Strings panel:

1. Select Window > Other Panels > Strings to open the Strings panel.
2. Click Import XML to show the Import XML dialog box.
3. In the Select a Language pop-up menu, select the language of the XML file you are importing, and click OK.
4. Navigate to the folder and XML file to import.

   The XML information is loaded into the column in the Strings panel for the language you selected in step 3.

   Note: Be sure to select the same language in Steps 3 and 4. Otherwise, you could, for example, import a French XML file into the column for German.

Regardless of where the XML file you imported was located, when you save, test, or publish the Flash document (FLA), a folder for each language in the Strings panel and an XML file for each language are created in the location indicated for publishing SWF files. For more information, see "Publishing Flash documents" on page 311. If no publish path is indicated, the folder and file are saved in the same folder in which the FLA file is located. The XML files generated by the Strings panel are always populated with the information in the Strings panel.

Creating documents with multilanguage text without using the Strings panel

You can create documents with multilanguage text without using the Strings panel. **Using the XMLConnector component to connect to external XML files**

You can use the XMLConnector component to connect to an external XML document in order to bind to properties in the document. Its purpose is to read or write XML documents using HTTP `GET` operations, `POST` operations, or both. It acts as a connector between other components and external XML documents. The XMLConnector communicates with components in your application using either data binding features in the Flash Professional authoring environment or ActionScript code. For more information, see "XMLConnector component (Flash Professional only)" in *Using Components*.

Using a Western keyboard to enter Asian characters on the Stage

With Flash MX 2004, you can enter Asian characters on the Stage using a standard Western keyboard by using Input Method Editors (IMEs). Flash supports more than two dozen IMEs.

For example, if you want to create a website that will reach a broad range of Asian viewers, you can use a standard Western (QWERTY) keyboard to create text in Chinese, Japanese, and Korean simply by changing the input method editor.

In previous versions of Flash, it was not possible to input Korean characters using a standard Western keyboard. With Flash MX 2004, you can enter text in Korean, Japanese, and Chinese characters simply by toggling the IME from Japanese and Chinese character input to Korean character input.

Note: This affects only text input on the Stage, not text entered in the Actions panel. This feature is available for all supported Windows operating systems and Macintosh OS X.

To toggle between Japanese and Chinese character input and Korean character input:

1. Select Edit > Preferences, and click the Editing tab in the Preferences dialog box.
2. Under Input Language Settings, select one of the following options:
   - Select Chinese and Japanese to input Chinese and Japanese characters from a Western keyboard. (This is the default setting, and it should also be selected for Western languages.)
   - Select Korean to input Korean characters from a Western keyboard.
3. Click OK.

Using ActionScript to load external files

If you have existing XML data you want to load, or prefer a different format for the XML file, instead of using the Strings panel, you can create a document containing multilanguage text by placing the text in an external text or XML file and loading the file into the movie clip at runtime, using the `loadVariables` action, the `getURL` action, the `LoadVars` object, or the `XML` object.

You should save the external file in UTF-8 (recommended), UTF-16BE, or UTF-16LE format, using an application that supports the format. If you are using UTF-16BE or UTF-16LE format, the file must begin with a BOM to identify the encoding format to Flash Player. For more information, see "Unicode encoding formats supported by Flash Player" on page 237.

Note: If the external file is an XML file, you cannot use an XML encoding tag to change the encoding of the file. You should save the file in a supported Unicode format. For more information, see "About encoding in external XML files" on page 237.

To include multilanguage text using an externally loaded file:

1. In the Flash authoring tool, create a dynamic or input text field to show the text in the document. For more information, see Chapter 6, "Working with Text," on page 105.
2. In the Property inspector, with the text field selected, assign an instance name to the text field.
3. Create a text or XML file that defines the value for the text filed variable.
4. Save the file in UTF-8 (recommended), UTF-16BE, or UTF-16LE format.

   If you are using UTF-16 format, make sure a BOM is included at the beginning of the file to identify the encoding:

   - For UTF-16BE, the first byte of the file should be OxFE, and the second byte should be OxFF.
   - For UTF-16LE, the first byte of the file should be OxFF, and the second byte should be OxFE.

   Note: Most text editors that can save files in UTF-16BE or LE automatically add the BOMs to the files.

5. Use one of the following ActionScript procedures to reference the external file and load it into the dynamic or input text field:

   - Use the `loadVariables` action to load an external file. For more information, see `loadVariables()` in *Flash ActionScript Language Reference*.
   - Use the `getURL` action to load an external file from a specified URL. For more information, see `getURL()` in *Flash ActionScript Language Reference*.
   - Use the `LoadVars` object (a predefined client-server object) to load an external text file from a specified URL. For more information, see "LoadVars class" in *Flash ActionScript Language Reference*.
   - Use the `XML` object (a predefined client-server object) to load an external XML file from a specified URL. For more information, see "XML class" in *Flash ActionScript Language Reference*.

Creating documents with multilanguage text using the #include action

You can create a document that contains multiple languages using the `#include` action.

You should save the text file in UTF-8 format. Save the file using an application that supports UTF-8 encoding, such as Dreamweaver.

You must include the following header as the first line of the file, to identify the file as Unicode to the Flash authoring tool:

```
//!-- UTF8
```

Note: Be sure to include a space after the second dash (-).

By default, the Flash authoring application assumes that external files that use the `#include` action are encoded in the traditional code page of the operating system running the authoring tool. Using the `//!-- UTF8` header in a file tells the authoring tool that the external file is encoded as UTF-8.

To include multilanguage text using the #include action:

1. In the Flash authoring tool, create a dynamic or input text field to display the text in the document. For more information, see Chapter 6, "Working with Text," on page 105.
2. In the Property inspector, with the text field selected, assign an instance name to the text field.
3. Create a text file that defines the value for the text field variable. Remember to add the header **//!-- UTF8** at the beginning of the file.
4. Save the file in UTF-8 format.
5. Use the `#include` action to include the external file in the dynamic or input text field. For more information, see `#include` in *Flash ActionScript Language Reference*.

Creating documents with multilanguage text using text variables

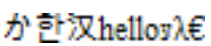
You can include Unicode-encoded contents in text variables using the syntax `\uXXXX`, where `XXXX` is the four-digit hexadecimal code point, or *escape* character, for the Unicode character. The Flash authoring tool supports Unicode escape characters through `\uFFFF`. To find the code points for Unicode characters, refer to the Unicode Standard at www.Unicode.org.

You can use Unicode escape characters only in text field variables. You cannot include Unicode escape characters in external text or XML files; Flash Player 6 does not recognize Unicode escape characters in external files.

For example, to set a dynamic text field (with the variable name `myTextVar`) that contains Japanese, Korean, Chinese, English, Hebrew, and Greek characters and the Euro sign, you can enter the following:

```
myTextVar = "\u304B\uD55C\u6C49hello\u05E2\u03BB\u20AC";
```

When the SWF file plays, the following characters appear in the text field:



For best results when creating a text field that contains multiple languages, make sure to use a font that includes all the glyphs your text needs. For more information, see "Using external text or XML files that are not Unicode encoded" on page 251.

Using external text or XML files that are not Unicode encoded

If you load external files into a Flash Player 7 application that are not Unicode-encoded, the text in the external files does not appear correctly when Flash Player attempts to show them as Unicode. You can tell Flash Player to use the traditional code page of the operating system that is running the player. To do this, add the following code as the first line of code in the first frame of the Flash application that is loading the data:

```
system.useCodepage = true;
```

Set the `system.useCodepage` property only once in a document; do not use it multiple times in a document to make the player interpret some external files as Unicode and some as other encoding because this can yield unexpected results.

If you set the `system.useCodepage` property to `true`, remember that the traditional code page of the operating system running the player must include the glyphs used in your external text file for the text to appear. For example, if you load an external text file that contains Chinese characters, those characters do not appear on a system that uses the CP1252 code page because that code page does not include Chinese characters. To ensure that users on all platforms can view external text files used in your Flash applications, you should encode all external text files as Unicode and leave the `system.useCodepage` property set to `false` by default. This causes Flash Player to interpret the text as Unicode. For more information, see `System.useCodepage` in *Flash ActionScript Language Reference*.

CHAPTER 14
Data Integration (Flash Professional Only)

Macromedia Flash MX Professional 2004 provides a flexible, component-based architecture and object model for connecting to external data sources, binding data to user interface (UI) components, and managing what data is displayed and how it's updated at the source.

The Macromedia website and Flash Help has many tutorials on creating rich Internet data applications in Flash. For downloadable examples and tutorials that use the data components, see "Additional resources" on page 255.

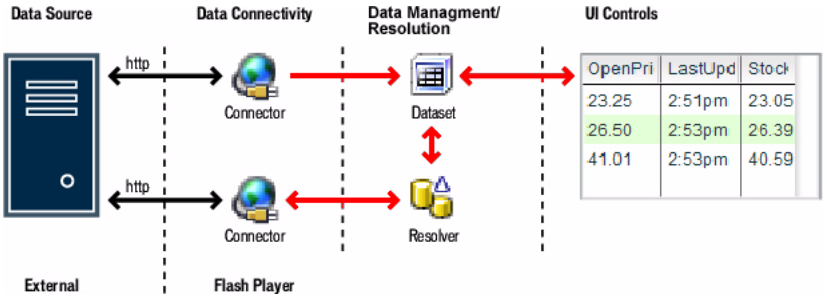
This chapter begins with an overview of data integration, provides a quick example you can walk through to become familiar with how data integration works, provides general workflows, and then explains data binding, which is the core functionality of the Flash data integration architecture, and the other layers in the Flash data integration architecture.

There are four main layers in the Flash data integration architecture:

- The data binding layer provides a way to map data elements to properties of Flash data components, which can then be mapped to UI components. In other words, you bind to a data source and then select the elements you need to display in your application and to update the source. Flash also integrates objects such as formatters and encoders to let you control how data is propagated and formatted among components. See "Data binding (Flash Professional only)" on page 259.
- The data connectivity layer provides connector components that let you connect to an external data source to send and receive data. You can connect to a variety of sources, such as web services and XML. For more information, see "Data connectivity (Flash Professional only)" on page 274.
- The data management layer provides a component that enables intelligent supervision of common data operations, such as editing, sorting, filtering, aggregation, and translation of changes. For more information, see "Data management (Flash Professional only)" on page 280.
- The data resolution layer provides resolver components that can translate changed data into a format that is consumable by an external data source. In addition, these components can accept and translate updates from an external data source so that they can be consumed by a Flash client. For more information, see "Data resolution (Flash Professional only)" on page 286.

When you integrate external data into a Flash application, you connect to the external data, select different elements of the data schema that you need for your application, and bind them to component fields within your application. You manage how the data is displayed in your application and how it's updated on the server.

The following image depicts the flow of data within a Flash application and identifies the different elements that comprise the Flash data architecture. Data binding is represented by the red arrows between the components. As shown in the diagram, you will need to set up data bindings between properties of UI controls and properties of a DataSet component; between the DataSet component and a connector component; between the DataSet component and a resolver component; and between a resolver and a connector component.



Typically, you add data components to the Stage in a Flash document. (See "Workflows for using the data components" on page 257 and in each component entry in Components Help.) The data components have no visual appearance in a runtime application. If you prefer, you can also create and access the data components through ActionScript code, although you may still need to perform some tasks through the Flash interface. To work with data binding classes in ActionScript instead of in the Flash interface, see "Making data binding classes available at runtime (Flash Professional only)" in *Using Components*.

The following table can help you decide what components you need to use in your Flash data application.

Data source	Use this connector	Use this resolver
web service/SOAP	WebServiceConnector WebService classes (not a component)	XUpdateResolver WebService classes (not a component)
XML document	XMLConnector	XUpdateResolver
SQL data	WebServiceConnector	RDBMSResolver

Flash is a client-side technology. To create a Flash application that integrates with a data source, you will need to implement server-side code as well. Building and exposing business logic on the server is the job of a server developer and is best implemented using products that are specifically designed for that task, such as Cold Fusion, J2EE Application Servers, and ASP.NET. For information on server-side tasks and other tasks that might best be addressed by a database administrator, see "Advanced topics in data integration" on page 289.

For more information, see the following topics in this chapter:

Additional resources

The following table outlines additional resources that are available for learning to use the data integration components in Flash.

Component	Data tutorials in Flash Help	Data tutorials on DevNet (www.macromedia.com/devnet/mx/flash/data_integration.html)
WebServiceConnector	Web Service Tutorial: Macromedia Tips	Tip of the Day, Part 2, www.macromedia.com/devnet/mx/flash/articles/tipoday_pt2.html Building a Google Search Application, www.macromedia.com/devnet/mx/flash/articles/google_search.html
XMLConnector	XML Tutorial: Timesheet	Timesheet Tutorial in Flash Help Bike Trips Sample, www.macromedia.com/devnet/mx/flash/articles/xmlconnector.html Data Integration Using ASP, www.macromedia.com/devnet/mx/flash/articles/flashpro_asp.html
XUpdateResolver	XUpdate Tutorial: Update the Timesheet	---
RDBMSResolver	---	Time Entry Application, www.macromedia.com/devnet/mx/flash/articles/time_entry.html Data Integration Using ASP, www.macromedia.com/devnet/mx/flash/articles/flashpro_asp.html Using the RDBMSResolver Component to Update a Database, www.macromedia.com/devnet/mx/flash/articles/delta_packet.html

Creating a simple application

The following example walks you through creating a simple data integration application, which can help you understand the concepts and steps involved.

In the example, you create a simple application that loads and displays a dinner menu. You load an XML file, which you'll use both as a data source and as a sample for the data source's schema. The UI consists of a data grid, into which the XML data is loaded, and a button that loads the data. Data binding is supported only between components that exist in Frame 1 of the main Timeline, Frame 1 of a movie clip, or Frame 1 of a screen; in this example, the components all reside in Frame 1 of the main Timeline.

To create the dinner menu application:

1. Copy the data source, an XML file called dinner_menu.xml, from the \Samples\HelpExamples\dinner_menu folder, and save it in a new folder, Dinner Menu, on your hard drive.
2. In Flash, create a new Flash document and save it as **dinner_menu.fla** in the Dinner Menu folder you created in step 1.
3. Create the user interface, which consists of two components—a button that triggers data retrieval and a data grid to display the data:
   a. Add a DataGrid instance named `menu_dg` to the stage with width **540** and height **240**.
   b. Add a Button component instance named `loadData` below the data grid labeled **Load Data**.
4. Add the data components—an XMLConnector component to connect to the dinner_menu.xml file and a DataSet component to bind that data to the data grid:
   a. Add an XMLConnector component instance named `xmlConn`.
   b. Add a DataSet component instance and name it `menu_ds`.

   The data components do not have to be on the Stage; they don't appear at runtime.
5. Set parameters for the XMLConnector component: select the XMLConnector component and, in the Component inspector, click the Parameters tab, enter `dinner_menu.xml` for the URL, and select `receive` for the direction. (Because the XML file is in the same folder as the FLA file, the fully qualified path is simply the XML filename.)
6. Load a sample of the data source's schema: click the Schema tab, select `results`, click Import a Schema from a sample XML file, and select the dinner_menu.xml file.

   The schema appears in the Schema tab.
7. Expose the XMLConnector's `array` property for data binding and bind it to the DataSet's `dataProvider` property. With the XMLConnector component selected, follow these steps:
   a. On the Bindings tab of the Component inspector, click the plus (+) sign and in the dialog box, select `food:array`.
   b. On the Bindings tab again, click Bound To, click the magnifying glass icon, select DataSet, and select `dataProvider:Array`.

   Each time you create a binding, you perform at least these two basic steps.

8. Next, populate the data grid with the XML data by binding the XML data—through the DataSet component— to the data grid. Select the DataSet component and click the Bindings tab. You see the binding to the `xmlConn` instance that you just added. Now, add two new bindings:
   a. Bind the DataSet's `dataProvider` property to the DataGrid's `dataProvider` property: click the plus (+) sign, select the `dataProvider:Array` property, click Bound To, click the magnifying glass icon, select DataGrid, then select the `dataProvider:Array` property. Select `out` for the direction.
   b. Bind the DataSet's `selectedIndex` property to the DataGrid's `selectedIndex` property: click the plus (+) sign, select the `selectedIndex:Number` property, click Bound To, click the magnifying glass icon, select DataGrid, then select the `selectedIndex:Number` property.
9. Set up the button to load data into the data grid. Add the following code to the first frame:

```
form = new Object();
form.click = function(eventObj){
  xmlConn.trigger();
}
loadData.addEventListener("click", form);
```

10. Save and test the application. Click Load Data. The data from the XML file is loaded into the DataGrid.

    You've just created your first data integration application, with data dynamically loaded from an XML file. To add more functionality to this application, see "Creating an indexed binding" on page 270.

Workflows for using the data components

This section provides a high-level overview of the steps required to create a Flash application that can dynamically interact with an external data source. You can find instructions and examples to complete each step throughout the rest of the chapter.

There are two general workflows: one for connecting to web services or XML documents as your data source, and one for connecting to an external database.

Workflow for data source from web services or XML documents:

1. Get the URL of your external data source:
   - A web service.
   - An XML document.
2. Add components to the Stage:
   - Add a connector component.
   - Add a DataSet component, which you will bind to your data source and UI components.
   - Add UI components that will display data to users, such as a DataGrid component.
   - Add a resolver component.

3. Set up the connector component:
   - Set component parameters.
   - Set component properties on the Schema tab.
4. Bind the connector component to DataSet component.
5. Set up the DataSet component:
   - Set component parameters.
   - Set component properties on the Schema tab.
6. Bind the UI component to the DataSet component.
7. Set up the resolver component:
   - Set component parameters.
   - Set component properties on the Schema tab.
8. Bind the resolver component to the DataSet component.
9. Add additional UI components and code for the resolver functionality (that is, for adding, editing, or deleting data records).
10. Bind UI components to resolver components.

Workflow for an external database (non-XML or not a web service):

1. Set up your data source; for example, in the ColdFusion environment, set up a ColdFusion DataSource component to connect to your data source.
2. Add components to the Stage:
   - DataSet component.
   - UI component for data display, such as DataGrid.
   - Resolver component.
3. Bind the DataSet component to the UI component for data display.
4. Set up DataSet component:
   - Set component parameters.
   - Set component properties on the Schema tab.
5. Set up a connection to your data; for example, you could set it up through a ColdFusion component with Flash Remoting services and your own ActionScript code.
6. Bind the resolver component to the DataSet component.
7. Set up the resolver component:
   - Set component parameters.
   - Set component properties on the Schema tab.
   - Write ActionScript code using methods of a resolver component class.
8. Add additional UI components and ActionScript code for the resolver functionality (that is, for adding, editing, or deleting data records). Bind UI components to resolver components.

Data binding (Flash Professional only)

Data binding lets you map the properties of one component to another component. A binding is simply a statement that says "When property X of component A changes, copy the new value to property Y of component B."

For rich internet applications, you can map data from external data sources to Flash components. The external data source is represented in your application by a component; items in the data source's schema are represented as properties of the component. You can define component properties to meet your business needs; these properties, which contain dynamic data that you want to manipulate, are referred to as *bindable* properties.

The most powerful use of data binding in Flash is to define the flow of data between UI components, data management components, and connector components that access external data sources such as web services, XML documents, and relational databases.

In the Flash interface, you bind data by using the Bindings and Schema tabs of the Component inspector. Although you need to understand how bindings and schemas work in Flash, your connector component is usually the first component you need to set up, because it brings in the schema for your data source; see "Data connectivity (Flash Professional only)" on page 274.

Data binding is supported only between components that exist in Frame 1 of the main Timeline, Frame 1 of a movie clip, or Frame 1 of a screen.

You can also create runtime bindings by writing ActionScript code. For more information, see "Data binding classes (Flash Professional only)" in *Using Components*.

A simple binding example

The following procedure provides a simple illustration of how data binding connects one UI component to another. In the example, the value properties of component instances `stepper1_nm` and `stepper2_nm` are bound to each other, and the value properties of `stepper3_nm` and `myInput_txt` are bound to each other. In a real-world application, you would most likely import a schema, define additional bindable component properties, and create multiple bindings between data components and UI components.

To connect UI components to create data binding:

1. Add a NumericStepper component to the Stage, and name it `stepper1_nm`.
2. Add another NumericStepper component, and name it `stepper2_nm`.
3. With `stepper1_nm` selected, open the Component inspector, and click the Bindings tab.
4. Click the Add Binding (+) button to add a binding.
5. In the Add Binding dialog box, select Value, and click OK.
6. In the Name/Value section at the bottom of the Bindings tab, click the Bound To item under Name, and click the magnifying glass icon across from the Bound To item under Value.
7. In the Bound To dialog box, select component `stepper2_nm` under Component Path, and click OK.

8. Select Control > Test Movie. Click the Up and Down buttons on component `stepper1_nm`.

   Each time you click the buttons on `stepper1_nm`, the `value` property of `stepper1_nm` is copied to the `value` property of `stepper2_nm`. Each time you click the buttons on `stepper2_nm`, the `value` property of `stepper2_nm` is copied to the `value` property of `stepper1_nm`.

9. Return to editing the application.

10. Add another NumericStepper component and name it `stepper3_nm`.

11. Add a TextInput component called `myInput_txt`.

12. Repeat steps 4-7 and bind the `value` property of `stepper3_nm` to the `text` property of `myInput_txt`.

13. Select Control > Test Movie. Type a number in the text input field, and press Tab.

    Each time you enter a new value, the `text` property of `myInput_txt` is copied to the `value` property of `stepper3_nm`. When you click the Up and Down buttons on `stepper3_nm`, the `value` property of `stepper3_nm` is copied to the `text` property of `myInput_txt`.

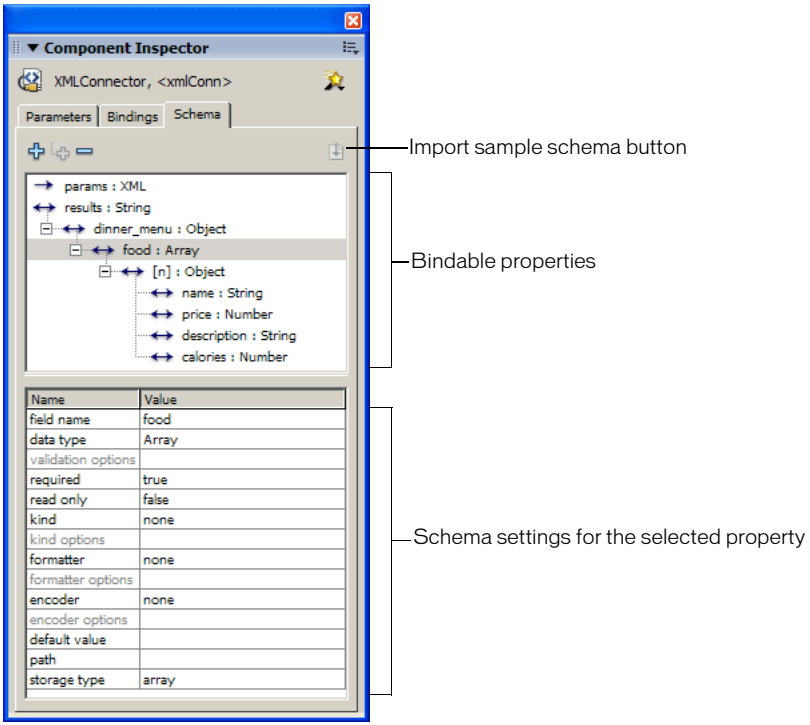
For more tutorials that show you how to create data bindings, see www.macromedia.com/devnet/mx/flash/data_integration.html.

Working with schemas in the Schema tab (Flash Professional only)

The Schema tab in the Component inspector lets you view and edit the schema for each data-related component in your application. The Schema tab lists the component's *bindable properties*, which are properties to which you can bind that commonly contain dynamic data. All components have properties, but by default, to reduce UI clutter, the Schema tab shows only properties that commonly contain dynamic data. (You can, however, bind to any property by either adding it to the Schema tab or using ActionScript code. For more information, see "Working with bindings in the Bindings tab (Flash Professional only)" on page 265.)

The Schema tab also lists properties' data types, their internal structure, and various special attributes. The data binding engine needs this information for each component to handle your data correctly.

The following illustration shows the Schema tab for the XMLConnector component used in "Creating a simple application" on page 256. The top pane shows the bindable properties for the `xmlConn` instance, with the `food:Array` property selected, and the bottom pane shows the settings for the `food:Array` property.



A component's schema describes the structure and type of data but is independent of how the data is actually stored. For example, the results from a WebServiceConnector component or an XMLConnector component could have identical schemas, even though the web service results are stored as ActionScript data structures (objects, arrays, strings, Boolean values, and numbers), and the XMLConnector component results are stored as XML objects. When you use data binding to access fields within a component's schema, you use the same procedure regardless of how the data is stored.

A component identifies which of its properties are bindable. These bindable properties appear in the Schema panel as top-level schema items (component properties). A component property can have its own internal schema that defines additional schema fields that can be bound to other component properties within your application; for example, when you introspect a WSDL for a WebServiceConnector component. The WSDL definition describes the parameters and the results for a web service. The WebServiceConnector component contains two bindable properties (`params` and `results`). When the WebServiceConnector component introspects the WSDL, Flash automatically creates the schema for the `params` and `results` properties so it mirrors the schema defined within the WSDL.

There are several ways to define the schema for a component. Here are the most common ways:

- For an XMLConnector component, you can import an XML sample file to define the schema. See "Connecting to XML data with the XMLConnector component (Flash Professional only)" on page 277.
- For a WebServiceConnector component, you can import the WSDL for a web service to define the schema. See "Connecting to web services with the WebService connector component (Flash Professional only)" on page 274.
- For a DataSet component, which is typically the intermediary component between your connector components and UI components, you define the schema using the Schema panel. See "Adding a component property to a schema" on page 262 and "Adding a schema field to a schema item" on page 263.
- For UI components, the schema is predefined within the component. You can modify the schema to create additional bindable properties, as shown in "Adding a component property to a schema" on page 262.

Adding a component property to a schema

You typically add component properties to a schema for the following reasons:

- To make an existing component property bindable. You can make any component property bindable if you add it to the schema.
- To define the fields of a DataSet component to describe expected data fields. Most commonly, you need to define the data type for an expected field, but there are numerous other properties you can set. For more information, see the examples in "Accessing the data" on page 284 and "Schema item settings" on page 289.

The following example illustrates how you make an existing component property bindable by adding the component property to the component's schema. In the example, you create an application that uses a CheckBox component to indicate whether a TextInput component is editable. Because the TextInput component's schema does not initially contain the `editable` property, you add the `editable` property to the schema to bind it to the CheckBox component.

To add a component property to a schema to make the property bindable:

1. Add a TextInput component and a CheckBox component to your application, and give them instance names.
2. Select the TextInput component, and click the Schema tab on the Component inspector.
3. Click the Add a Component Property (+) button at the upper left of the Schema tab to add a component property.
4. In the Schema Attributes pane (the bottom pane of the Schema tab), enter **editable** for the field name value and select Boolean for the data type value.
5. Click the Bindings tab, and click the Add Binding (+) button to add a binding.
6. In the Add Binding dialog box, select the `editable` property, and click OK.
7. In the Binding Attributes pane at the bottom of the Bindings tab, click the Bound To item under Name, and click the magnifying glass icon across from the Bound To item under Value.

8. In the Bound To dialog box, select the CheckBox component under Component Path, and click OK.
9. Select the Checkbox component on the Stage, and click the Parameters tab in the Component inspector.
10. Select Control > Test Movie. To test the functionality, type a value into the TextInput component and then deselect the CheckBox component. You should now be unable to enter text into the TextInput component.

Adding a schema field to a schema item

When you use a DataSet component, you manually enter the schema for the component. You might need to add schema items, which are essentially component properties (see "Adding a component property to a schema" on page 262). You may also need to add additional fields within a schema item to provide a deeper level of bindable detail. For more information, see "Schema item settings" on page 289.

To add a schema field to a schema item:

1. In the Schema tab, select the schema item to which you want to add a field.
2. Click the Add a Field Under the Selected Field (+) button.

   A new field is added as a subfield of the selected property.
3. In the Schema Attributes pane, enter a value for Field Name. Fill in the other attributes as appropriate.

   There are three possible scenarios based on the type of schema item:

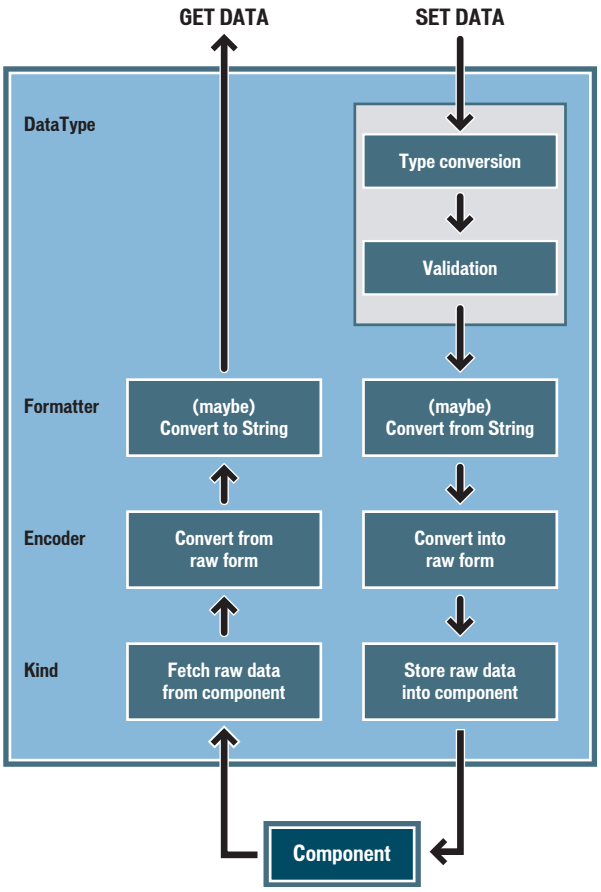
   - Schema item of type Object, which can have subfields, attributes, or both. Attributes are preceded with @ in the list.
   - Schema item of type Array, which has one subfield called [n] representing the index of the array, which can be of any type (including Object, String, and so on).
   - Schema item of other types (such as Boolean, String, Number), which don't have subfields but can have attributes. Attributes are preceded with @ in the list.

About handling data types in data binding (Flash Professional only)

Your data source's schema is represented on the Schema tab in the Component inspector. Every item in the schema has many attributes that you can configure in the lower pane of the Schema tab. In particular, four attributes control the handling of data types as data flows in and out of Flash applications. These four attributes are Data Type, Encoder, Formatter, and Kind.

You might not need to change the settings of these attributes from the default values. However, in situations where you're working with complex data types, you might need to change the values of these attributes so Flash receives and outputs data in the correct format. See "When to edit schema item settings" on page 300.

The following illustration shows the runtime process of the data binding engine. The four attributes that handle data types are shown in the illustration and discussed in the following text.



Kind When Flash wants to get data from a component, the data is fetched from the component according to the Kind setting. At this point, the data is in whatever format the component provides (the raw form of the data). For example, the XMLConnector component always provides data as a string, the NumericStepper component provides data as a Number, and so on.

Encoder The encoder's job is to convert this data to an ActionScript data type. For example, the string data that you get from an XML document can represent a date or a number. If data binding needs the data in string form (because it is being assigned to a text component, for example) the formatter does this conversion. If there are several bindings from a field, the formatter is used only for those bindings that are assigning to a field whose type is String.

Data type and formatter When you want to set data into a component, data binding first needs to convert the data to an ActionScript data type, which is a form that the component can read; this conversion is automatic, depending on the Data Type setting. If the data is a string and a Formatter setting exists, then the formatter converts the data from string to the specified ActionScript data type. The Data Type setting also controls whether the data binding engine inspects the data to see if it's valid and causes events to be generated accordingly. The encoder then converts the data from the ActionScript-readable form to raw form, and the kind then finally passes the data to the component.

The processing handled by these four attributes occurs when the data field is accessed with data binding. It is possible to directly access a component property from your ActionScript code, but when you do this, you're working with the raw value of the data, not the data value that results from the action of data types, encoders, formatters, and kinds. For more information, see the "DataType class (Flash Professional only)" in *Using Components*.

In many cases, you won't need to edit the settings that are in the bottom pane of the Schema tab. The following guidelines specify when to change the schema item settings from their default values:

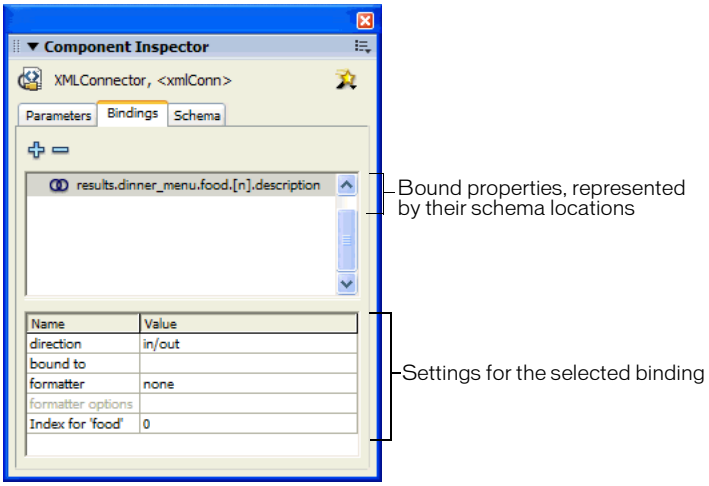
- You always need a kind. The default value for the kind setting is `none`, which is equivalent to the Data kind.
- You need an encoder when the component does not provide the data in the form you want. The most common case is the XMLConnector component, or any other component whose properties are XML data. This is because XML stores all data—including numbers, dates, and Boolean values—as strings. If you want to use the actual data instead of its string representation, you use an encoder.
- You need a formatter when you want to control how the data is converted to a string, usually for display purposes.
- You need a data type when you want data validation to occur, you want better conversion for certain data types, or both.

For more information on these schema item settings, see "Schema item settings" on page 289.

Working with bindings in the Bindings tab (Flash Professional only)

Once you have imported and defined schemas for your data components, as described in "Working with schemas in the Schema tab (Flash Professional only)" on page 260, you can start adding bindings. You use the Bindings tab to add and remove bindings to and from components and their properties. All the bindings for a component appear here.

The following illustration shows the Bindings tab. The top pane lists the properties exposed for binding, represented by their schema location, of the component that's selected on the Stage and contains Add Binding (+) and Remove Binding (-) buttons. The bottom pane shows information about settings for the selected property, such as what it's bound to and in which direction it's bound.

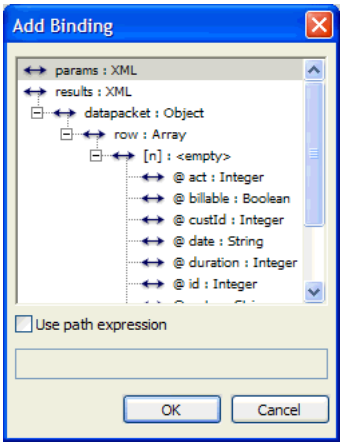


To walk through the steps of creating bindings, see "A simple binding example" on page 259. The following topics describe each step of creating bindings in more detail:

- "Adding a binding" on page 267
- "Configuring bindings" on page 268
- "Defining what to bind to" on page 269
- "Creating an indexed binding" on page 270

Adding a binding

To add a binding, click the Add Binding (+) button on the Bindings tab. The Add Binding dialog box appears.



This dialog box shows all the schema items (properties) for the selected component. You use this dialog box to select which property you want to expose for binding. Component properties appear as root nodes within the schema tree. An arrow icon represents whether a schema item has read/write access, as follows: a right-pointing arrow represents a write-only property, a left-pointing arrow represents a read-only property, and a bidirectional arrow represents a read-write property. (See "Configuring bindings" on page 268.)

To walk through the steps of creating a binding, see "Creating a simple application" on page 256, which creates a simple data application, or "A simple binding example" on page 259, which demonstrates how bindings connect two UI components.

In general, follow these steps to add a binding:

1. Select the component on the Stage for which you want a binding.
2. In the Component inspector, click the Bindings tab.
3. Click the Add Binding button. The Add Binding dialog box opens.
4. Select the property for which you want to add a binding.
5. In the bottom pane of the Bindings tab, click Bound To. The value field becomes editable.
6. Click the magnifying glass icon in the field and select the component path and schema location to bind to. See "Defining what to bind to" on page 269.
7. In the bottom pane of the Bindings tab, click Direction and select the appropriate value from the pop-up menu. See "Configuring bindings" on page 268.
8. Repeat the steps for additional components.

The schema for a component defines which schema items are bindable. However, you might need add a binding for a schema item that is not identified in the data source's schema. You can do this by selecting the Use path expression option. See "Adding bindings using path expressions".

Configuring bindings

When a property is selected on the Bindings tab, you can further define it using the options located in the bottom pane of the Bindings tab. You can specify information such as Direction and Bound To, which you'll commonly need to specify, as well more complex properties such as Formatter and Formatter Options:

Direction Shows a list of directions that can be set for a binding. You need to select a value from the list:

- In: The selected schema item is the destination of a binding. It receives a new value when the other end of the binding changes. On the Schema tab, in is represented by a left-pointing arrow:
- Out: The selected schema item is the source of a binding. Whenever its value changes, the value is copied to the other end of the binding. On the Schema tab, out is represented by a right-pointing arrow:
- In/Out: New data values are copied when either end of the binding changes value. On the schema tab, in/out is represented by a two-headed arrow:

Bound To Identifies the destination schema item (another component's property) to which this schema item is bound. You need to specify this value. See "Defining what to bind to" on page 269.

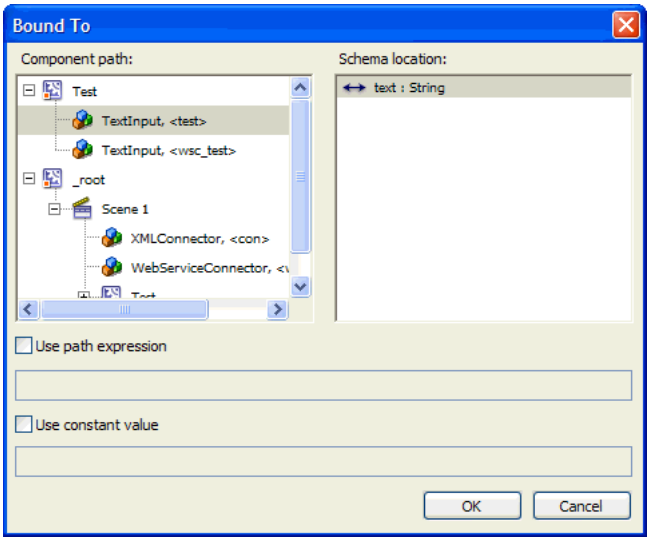
Formatter Shows a list of available formatters that determine how to display this binding. For more information, see "Schema formatters" on page 297.

Formatter Options Shows the Formatting Options dialog box. The settings in this dialog box are used at runtime to control formatting of data assigned from this schema item to the destination schema item that is defined in the Bound To property. These settings override the default formatting settings for the source schema item. See "Schema formatters" on page 297.

Index For If you create a binding for a schema item that is defined as a field of an object contained within an array, you must specify an index for the array. See "Creating an indexed binding" on page 270.

Defining what to bind to

When you expose a component property for binding, you need to define what to bind the property to. The Bound To dialog box appears when you click Bound To in the Binding Attributes pane of the Bindings tab. The Bound To dialog box includes the Component Path pane and the Schema Location pane.



The Component Path pane shows a tree of components that have properties to which you can bind. The tree is based on the current Stage editing environment:

- If the Stage shows the contents of the document root, a single component path tree appears relative to the document root.

  Note: Component instances are displayed only if they exist in Frame 1 of the edited document root or in Frame 1 of any screen/clip whose instance exists in the edited document root. This pane shows only components, not text fields.

- If the Stage shows the contents of a movie clip being edited from the library, two component path trees appears. The first appears from the root of the symbol being edited, and the second appears from the document root, allowing bindings to instances within the document.

  Note: Bindings to this second component tree do not appear in the Bound To instances when they are selected. They appear only as bindings from the Bound From component instance.

The Schema Location pane shows the schema tree of the component selected in the Component Path pane. This is the same information that appears in the Schema Tree pane of the Component inspector Schema tab.

You can use a dynamic value or a constant value for the Bound To property.

To use a dynamic value for the Bound To property:

1. Select a component in the Component Path pane.
2. Do one of the following actions to select a schema item for the data:
   - Select a schema item using the Schema tree located within the Schema Location pane.
   - Select Use Path Expression, select a component property from the schema tree, and enter a path expression. For more information, see "Adding bindings using path expressions" on page 302.

To use a constant value for the Bound To property:

- Select Use Constant Value, and enter a constant value, such as `3`, a string, or `true`. You can use any value that is valid for the schema item. When you use a constant value, the selected component path, schema location, and path expression are ignored. You can bind to a constant value only when the Direction attribute for the binding is set to In.

Creating an indexed binding

In the example application created in "Creating a simple application" on page 256, the data grid displays the dinner menu. The description of each food item, however, is too long to fit in the data grid. Ideally, the user could click an item in the data grid and read the full description of a food item, perhaps in a text box below the data grid. To accomplish this, you would create an indexed binding to the data array.

This section shows you how to create an indexed binding to connect a field in your data source with the selected index of another component. The most common use for an indexed binding is to the `selectedIndex` property of a UI element. When you create a binding to the index of an array, a setting for its value is dynamically added to the Schema Attributes pane; you use this setting, the `Index for` field, to specify to what to bind the index.

Note: If a schema item location includes several array references such as `"foo/bar[]/abc[]/def[]"`, three `index for` settings are dynamically added to the Schema Attributes pane—one for each array that needs to be indexed.

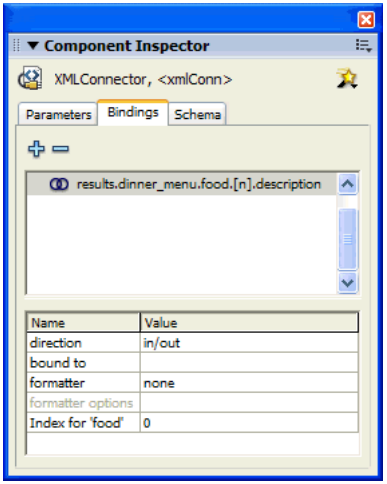
In the following example, you add a text box to display the full description of the food item when a user clicks on the item in the data grid.

To create an indexed binding:

1. If you haven't already done so, create the example application shown in "Creating a simple application" on page 256.
2. Drag a TextArea component to the Stage and name it `myTextArea`.

3. Select the `xmlConn` instance, click the Bindings tab, click the + symbol, and select the `description:String` property, which is in the `food` array.

   Notice that on the Bindings tab, the attribute `Index for 'food'` is dynamically added, as shown in the following image; you'll fill in this value in a later step.

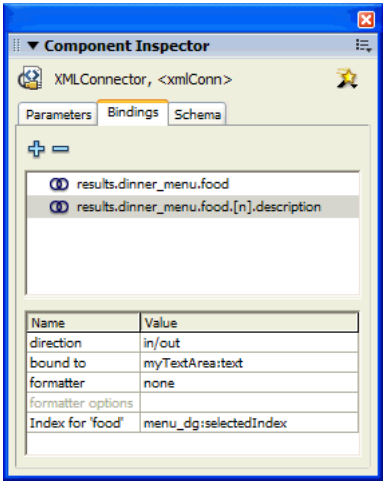


4. With the `results:dinner_menu:food.[n].description:String` field selected on the Bindings tab, click Bound To, click the magnifying glass icon, select `myTextArea`, and select the `text:string` property.

   The text area will be populated by the `description` property of the `food` array.

   Next, you define the index value for the `food` array, so that when the user clicks a different item in the data grid, the correct description populates the text box.

5. Click `Index for 'food'`, click the magnifying glass icon, deselect Use Constant Value, select the `menu_dg` DataGrid instance, and select `selectedIndex:Number`. The settings for the indexed binding appear in the Bindings tab, as shown in the following image:



6. Next, set the DataGrid index default value to 0 to make it available for data binding: select the `menu_dg` instance, click the Schema tab, select `selectedIndex:number`, and in the Default Value field in the lower pane, type `0`.
7. Save and test the application. Click Load Data, then click different items in the data grid. The text area updates with the detailed description for each food item. Each time the user selects a new item in the data grid, the index of the array is updated to show the data associated with the new item.

   Note: The `index for` property appears only in the Binding attributes pane for a schema item that is the field of an object within an array.

Sometimes you might need to manually define a schema that identifies a schema item as a field of an object contained within an array. In the following example, the id, billable, rate, and duration schema fields are all considered attributes of an object contained within the row array:

```
results : XML
   datapacket : Object
      row : Array
         [n] : object
            @id : Integer
            @billable : Boolean
            @rate : Number
            @duration : Integer
```

If a binding is created for any of these items, an `index for 'row'` property appears in the Binding Attributes pane, so that an index can be specified for the row array. Flash uses the [n] schema field to identify this type of relationship. Therefore, you might need to duplicate this entry if you are manually creating a schema. To do this, you add a new schema field under the `row : Array` node and set Field Name for the schema field to `[n]`. The compiler reads this value and creates an `index for` property if it is used within a binding.

About debugging data binding and web services (Flash Professional only)

Data binding is a series of actions that occur in response to events, such as the following:

- The data of a component property changes.
- A web service call is completed.
- An XML document is fetched.

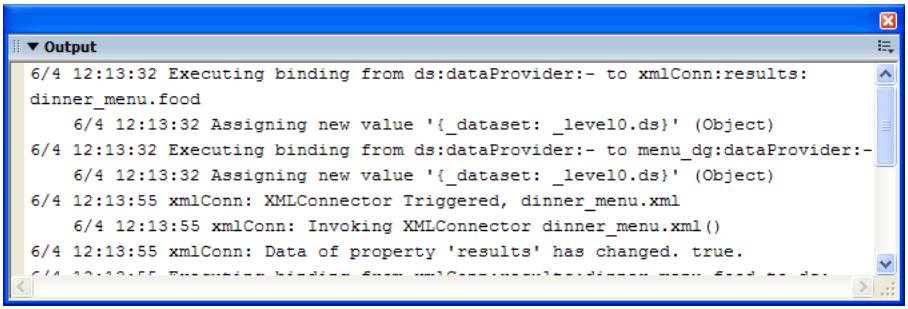
You can create a log of all actions that are performed by data binding or web services. To create the log, create a new Log object by adding the following code to the first frame in your Flash document:

```
_global.__dataLogger=new mx.data.binding.Log();  //to enable trace log
```

To turn the trace log off, use the following code:

```
_global.__dataLogger=null; //disable trace for binding.
```

When you run an application that turns the trace on, a detailed log of data binding and web services events and actions appears in the Output window. The following image shows the log for the application created in "Creating a simple application" on page 256, when the code to enable the trace log is added to the first frame of the application:



The following list describes the types of things reported:

- Executing bindings
- Calling web service methods
- Fetching XML documents
- Status and result events from WebService and XML components
- Valid and invalid events from validated data fields
- A variety of errors, invalid settings, and so on

By running your application and then examining the log, you can often find out why things are not working as expected. Sometimes an error is explicitly reported—for example, a missing web service parameter. Sometimes the data is bound to the wrong component or to no component and so on. If you find that there is too much information in the log, clear the Output window by selecting Clear from the context menu, to keep the log as short as possible.

For more information, see the "Log class (Flash Professional only)" in *Using Components*.

Data binding in Flash Player 7 versus Flash Player 6

Bindings between components are executed based on default component events (for example, a binding between the `selectedIndex` of a DataGrid and a DataSet is executed whenever a new record is selected in the DataGrid or DataSet. After the event is generated, the binding is queued to be executed as soon as possible. This action depends on your version of Flash Player. If you publish to Flash Player 7, the binding happens immediately. If you publish to an earlier version of Flash Player, the binding is queued to the beginning of the next frame.

However, the DataSet component works only in Flash Player 7. Queuing bindings to the next frame can potentially cause issues with components, such as the DataSet, that provide their own events for accessing data that may become out of sync with data binding. To avoid these issues, Macromedia recommends that you publish to Flash Player 7 when using the DataSet component.

Data connectivity (Flash Professional only)

You use the connector components in Flash to connect to your data source. The schema for your data source is mapped to properties of a connector component. A typical application might contain several connector components for retrieving or updating data, or both.

Before you can create data bindings, you must either set up a connector component on the Stage or create the proper mappings in ActionScript using the WebServiceConnector component class. However, it is useful to first understand how data bindings in Flash work; see "Data binding (Flash Professional only)" on page 259.

Note: External data refers to any data that is accessible through HTTP.

Flash comes with the following connector components:

- The "WebServiceConnector component (Flash Professional only)", which lets you connect to the WSDL URL of a web service.
- The "XMLConnector component (Flash Professional only)", which lets you connect to any external data source that returns XML through HTTP (such as JSP, ASP, Servlet, or ColdFusion).

In addition to, or instead of, using these connector components, advanced developers and database administrators can use the WebServices classes to write ActionScript code that accesses remote procedure calls exposed by a server using Simple Object Access Protocol (SOAP). For more information, see "Web service classes (Flash Professional only)" in *Using Components*.

Note: The WebService classes are accessible only through ActionScript code and are common to various Macromedia products. The WebServiceConnector component has an API that is unique to Flash MX 2004 and lets you access the component's methods, properties, and events through the visual interface.

To help you consider what kind of connectivity architecture you should implement, see the following DevNet articles: "Choosing Between XML, Web Services, and Remoting for Rich Internet Applications" at www.macromedia.com/devnet/mx/flash/articles/ria_dataservices.html and "Getting a Handle on Web Services" at www.macromedia.com/devnet/mx/flash/articles/flmxpro_webservices.html.

Connecting to web services with the WebService connector component (Flash Professional only)

The WebServiceConnector component lets you introspect, access, and bind data between a remote web service and your Flash application. A single instance of a WebServiceConnector component can be used to make multiple calls to the same operation. To call more than one operation, use a different instance of a WebServiceConnector component for each operation. For example, you would use one instance to connect to a DataSet component and another instance to connect to a resolver component, as shown in the illustration in the overview at the beginning of this chapter.

To use the WebServiceConnector component, you need to load the web service's schema into the WebServiceConnector component. A web service's schema is defined by a Web Service Description Language (WSDL) file. The WSDL file, which is accessible through a URL, specifies a list of operations, parameters, and results that are exposed by the web service. Once the schema is loaded, you can proceed to add data bindings.

You can load and view the schema of any web service by entering the URL into the `WSDLURL` parameter of a WebServiceConnector component instance.

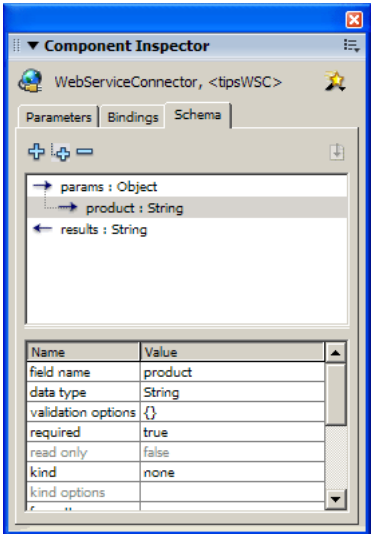
The following example demonstrates how to load and view the schema for a web service that provides helpful tips for different products. You add a WebServiceConnector component instance on the Stage, specify the web service to use, and view the web service's schema on the Schema tab of the Component inspector.

Note: This example requires an active Internet connection because it uses a public web service. If you use a web service in your application, the web service must be located in the same domain as the SWF file for your application so the application can work in a web browser. For more information, see "About data connectivity and security in Flash Player" on page 279.

1. Drag a WebServiceConnector component to the Stage and name it `tipsWSC`.
2. In the Component inspector, click the Parameters tab, if not already selected.
3. Select the `WSDLURL` parameter, and type the following URL:

   `http://www.flash-mx.com/mm/tips/tips.cfc?WSDL`

   When you specify a web service for a WebServiceConnector component in this way, it is automatically added to the Web Services panel and is available to any application you create.
4. Select Operation, and select the `getTipByProduct` method.
5. Click the Schema tab and view the auto-generated schema for the web service:



The Schema tab displays a schematic representation of the service that you are calling. The parameters and results structure are defined within the schema. The Tips schema states that the service expects one String parameter, `product`, when it is called; this is the write-only input, as indicated by the right-pointing arrow. The service returns a string as the result of the call; this is the read-only output, as indicated by the left-pointing arrow.

Once the web service's schema is brought into the Schema tab, the items identified within the schema can now be bound, using the Bindings tab, to a variety of UI controls to let users input values for the parameters and to get back and display the results of the web service. To see this web service in action, see the Tips application in the Samples/HelpExamples/tips folder. For information on data binding, see "Data binding (Flash Professional only)" on page 259 and "Working with bindings in the Bindings tab (Flash Professional only)" on page 265.

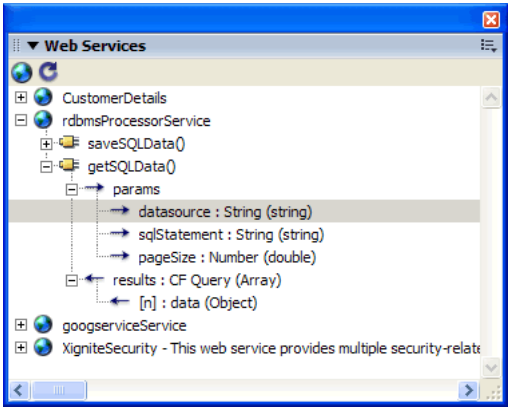
For a common workflow and information on the properties, methods, and events of the WebServiceConnector component, see "WebServiceConnector component (Flash Professional only)" and "Using the WebServiceConnector component (Flash Professional only)" in *Using Components*.

Using the Web Services panel

You can view a list of web services, refresh web services, and add or remove web services in the Web Services panel (Window > Development Panels > Web Services). When you add a web service to the Web Services panel, the web service is then available to any application you create. When you drag a WebServiceConnector component onto the Stage and specify a value for the WSDLURL parameter, that web service is automatically added to the Web Services panel.

You can use the Web Services panel to refresh all your web services at once by clicking the Refresh Web Services button. If you are not using the Stage but instead are writing ActionScript code for the connectivity layer of your application, you can use the Web Services panel to manage your web services.

The following illustration shows the Web Services panel, to which several web services have been added. A web service is represented by the planet icon, and its operations appear in the tree.



To add, edit the name of, or remove a web service:

1. Click Define Web Services (the planet icon at the top of the panel).
2. To add a service, click Add Web Service, and enter the URL of the web service. Double-click an existing web service to edit its name, or select a service and click Remove to remove it.

   If you want to edit a WebServiceConnector component's schema, you can edit it from the Schema tab of the Component inspector.

Note: Access to a web service (as with any external data) is subject to Flash Player security features. For more information, see "About data connectivity and security in Flash Player" on page 279.

Connecting to XML data with the XMLConnector component (Flash Professional only)

The XMLConnector component lets you access any external data source that returns or receives XML through HTTP. A single instance of an XMLConnector component can be used to make multiple calls to the same operation. To call more than one operation, use a different instance of an XMLConnector component for each operation. For example, you would use one instance to connect to a DataSet component and another instance to connect to a resolver component, as shown in the illustration in the overview at the beginning of this chapter.

To use the XMLConnector component, you load a sample of your XML document's schema into the component. The schema is the structure of the XML document that identifies the data elements in the document to which you can bind.

To load the schema, you import a sample of the XML data to which you're connecting. You can either use an actual sample of real data or, if you know XML scripting, create a sample yourself. You import that sample XML file using the Component inspector.

Be sure that the sample you use contains all the elements you want for data binding and accurately represents the real data. Different XML structures result in different schemas. For example, if your sample contains an array with only one item, Flash won't know that you need an index for that array. The array needs to contain at least two items.

To import a sample schema:

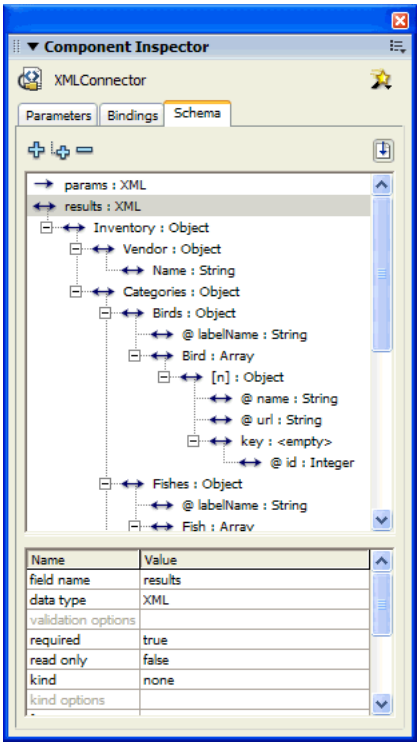
1. Locate the XML file to use as a sample.
2. Drag an XMLConnector component to the Stage.
3. Click the Parameters tab in the Component inspector and for the URL parameter, specify the fully qualified name of the XML data source.
4. Click the Schema tab in the Component inspector and select `params` or `results`, as appropriate. Select `results` if the XML sample represents the schema of the results of a call to the data source.
5. Do one of the following to import the schema:
   - Click the Import Sample Schema button in the upper right corner of the Schema tab.
   - Click the options menu control in the upper right corner of the Component inspector and select Import XML Schema from the menu.

6. In the Open File dialog box, select the file that you want to use as a sample, and click Open.

   The schema appears in the Schema tab. You can now create bindings between elements of your XML document and other component properties within your application.

Note: Some XML documents may have a structure that Flash MX cannot represent; for example, elements that contain text and child elements mixed together.

The following illustration shows the schema for a file named Animals.xml:



The schema tab displays a schematic representation of the structure of the XML file. It says that the `results` property of the XMLConnector component is an XML object. The root element of that object is called Inventory, which contains the elements Vendor, Categories, and so on. The Vendor element contains a single element called Name, which is a string. The Categories field contains an element called Birds, which contains the attribute `labelname`. The Birds element also contains an array of objects called Bird. Each of these objects has two attributes: `name` and `url`. It also contains a single element named key, which contains the attribute `id`. The index for the Bird array is represented by the `[n]` field.

The String and Integer fields can be bound to UI components. The Array field Bird can be bound to a DataSet component or to list-based UI components such as List, DataGrid, or ComboBox, which all use the data provider interface. Or, you can directly bind UI components to fields within certain records of the array, as shown in the example application in "Creating an indexed binding" on page 270.

A typical workflow for an application that works with data would include binding an array from the XMLConnector component to the DataSet component's `dataProvider` property. Or, you can directly bind UI components to fields within certain records of the array, as shown in the example application in "Creating an indexed binding" on page 270. In this scenario, the data set could be used to manage the data. The fields within the data set could then be mapped to any of the UI components using data binding.

For more information on the XMLConnector component, including its properties, methods, and events, see "XMLConnector component (Flash Professional only)" in *Using Components*. For a common workflow using this component, see "Using the XMLConnector component (Flash Professional only)" in *Using Components*.

You can also read the following tutorials on Macromedia DevNet: "Bike Trips Sample" at www.macromedia.com/devnet/mx/flash/articles/xmlconnector.html and "Data Integration Using ASP" at www.macromedia.com/devnet/mx/flash/articles/flashpro_asp.html.

About data connectivity and security in Flash Player

Many developers are interested in using an industry standard such as SOAP web services as the data-exchange mechanism between their client and server. One reason this approach is gaining favor is the increasing number of popular servers that support exposure of logic using SOAP.

There may be cases where you want the client software to use web services that are published by third parties or hosted on servers that fall outside the Flash Player sandbox. Access to external data through any connector component is subject to the sandbox security model in Flash Player, for all Flash applications that run in a web browser. The sandbox security model restricts a Flash document from accessing data from any domain other than the one in which it originated (this includes public web services). There are a couple of ways to accomplish what you want to do, while still preserving the user security and privacy that the Flash Player sandbox provides:

- Create a policy file that is hosted on the server containing the web service to be used. For more information, see "Flash Player security features" in *Using ActionScript in Flash* and the security tech note 14213 at www.macromedia.com/support/flash/ts/documents/loadvars_security.htm.
- Create an intermediary object that resides on the server to act as a bridge between your client and the public services you want to use. This approach offers several advantages:
  - Public web services can be aggregated. With this approach you can provide fail-over safety and load balancing when a request is made for data.
  - You can control the flow of data in your application. If the web service goes away or the URL is down, you can decide how to respond.
  - Data can be optimized. Multiple requests can be cached.
  - You can have custom error handling. You can determine what errors to send back to the client.
  - Data can be manipulated, converted, or combined. You can pull data from several sources and return one data packet with the combined information.

Many of the SOAP-based applications that you build will use private web services hosted on your server. After you determine the best way to implement and expose your own web services, it is easy to make public web services available to your client application. When you are in control of the server, you can offer a complete solution. The server is the ideal place for business logic that can determine the best way to respond to requests for data and the results that should be sent back to the client. This is also the most secure way to build an application. The server can provide additional processing to make sure that users have access only to certain services as well as protect the client from making calls to malicious services that can return bad data.

For more information, see the DevNet article "Getting a Handle on Web Services" at www.macromedia.com/devnet/mx/flash/articles/flmxpro_webservices.html.

Data management (Flash Professional only)

You use the DataSet component for applications that handle managed data. The term *managed data* refers to the ability to perform advanced operations on a local cache of data, including multiple sorts, filters, finds, and offline caching. A managed data solution requires more setup but gives you greater control over your data. In general, you should use a managed data approach for the following scenarios:

- You need to apply multifield sorts, filters, or ranges to your data.
- You are building an application that provides the ability to work offline (changes to the data are cached offline and can be applied at a later time).
- You want to receive changes from the server and apply them to your local cache of data.
- You want to create a custom transfer object implementation to complement a business class on the server.
- You plan to send updates back to an external data source using the built-in features of the DataSet and resolver components (such as automated tracking of changes to your data that can be converted into multiple formats).

For more information, see "Managing data with the DataSet component (Flash Professional only)" on page 281.

If your application displays dynamic read-only data, you can use a simpler approach that does not use the DataSet component. You would instead bind the results of a connector component directly to UI components within your Flash document.

The DataSet component uses functionality in the DataBinding classes. If you intend to work with the DataSet component in ActionScript only, without using the Binding and Schema tabs in the Component inspector to set properties, you'll need to import the DataBinding classes into your FLA file and set schema properties in your code. For more information, see "Making data binding classes available at runtime (Flash Professional only)" in *Using Components*.

For a tutorial that uses the DataSet component, see the DevNet article "Flash Data Integration Using Microsoft Active Server Pages (ASP)" at www.macromedia.com/devnet/mx/flash/articles/flashpro_asp.html.

The DataSet component works only with Flash Player 7.

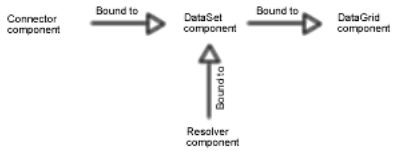
Managing data with the DataSet component (Flash Professional only)

The data structure that is fundamental to data-driven applications is a table with rows and columns, or fields. To expose the fields of the current row in the table, you must define properties of a DataSet component on the Schema tab. (For an example, see the design time example in "Accessing the data" on page 284.)

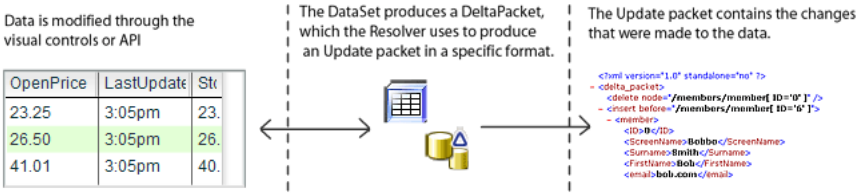
Once you have specified a schema for the DataSet component, you typically create the following bindings to or from a DataSet component:

- Bind the results of a connector component to fields of the DataSet component.
- Bind fields of the DataSet component to properties of UI components within your Flash document.
- Bind the DeltaPacket property of a resolver component to the DeltaPacket property of a DataSet component.

The following diagram illustrates the data binding that typically is needed when you use a DataSet component.



The DataSet component is used to hold and organize your data; you must use data bindings and write ActionScript code to handle updates. Changes that are made to your data through UI components can be tracked and used to generate a DeltaPacket, an object produced by the DataSet component that contains a list of changes made to data at runtime. A resolver component can then manipulate the DeltaPacket into a specific format for use by external data sources. Using the `logChanges()` method of the DataSet component, you can track both changes made to the data and methods called.The following illustration shows the flow of data through a UI component, DataSet and Resolver component, and the DeltaPacket object produced.



For a common workflow and information on how you use the methods, properties, and events of the DataSet component to manage your data, see "Using the DataSet component", "DataSet class (Flash Professional only)", and the "DeltaPacket interface (Flash Professional only)" in *Using Components*.

The DataSet component uses functionality in the DataBinding classes. If you intend to work with the DataSet component in ActionScript only, without using the Binding and Schema tabs in the Component inspector to set properties, you must import the DataBinding classes into your FLA file and set schema properties in your code. For more information, see "Making data binding classes available at runtime (Flash Professional only)" in *Using Components*.

The DataSet component works only with Flash Player 7.

For more information on working with data in the DataSet component, see the following topics:

- "About loading data into the DataSet component" on page 282
- "Accessing the data" on page 284

About loading data into the DataSet component

To load data into the DataSet component, you edit the schema for the DataSet and create data bindings that can be done either in ActionScript or on the Bindings tab of the Component inspector. You need to edit the schema, in most cases, so that data appears correctly in your application. For information on editing schema, see "Adding a component property to a schema" on page 262 and "Adding a schema field to a schema item" on page 263. You can create bindings for the DataSet component in two ways:

- An array of objects bound to the `DataSet.items` property (see `DataSet.items` in Using Components).
- An object bound to the `DataSet.dataProvider` property. This object should implement the DataProvider interface; see `DataSet.dataProvider` property and "DataProvider API" in *Using Components.*

The objects can be sophisticated client-side objects that mirror their server-side counterparts, or in their simplest form, a collection of anonymous objects with public properties representing the fields within a record of data.

The DataSet component uses functionality in the DataBinding classes. If you intend to work with the DataSet component in ActionScript only, without using the Binding and Schema tabs in the Component inspector to set properties, you'll need to import the DataBinding classes into your FLA file and set schema properties in your code.

The following examples show different ways you can load objects into the DataSet component, using either ActionScript code or the Component inspector. The examples assume that you have specified a schema for the DataSet component on the Schema tab first; see the design-time example in "Accessing the data" on page 284.

Anonymous objects The following ActionScript example assigns an array of 100 anonymous objects to the `items` property of the `myDataSet` instance of the DataSet component. Each object represents a record of data.

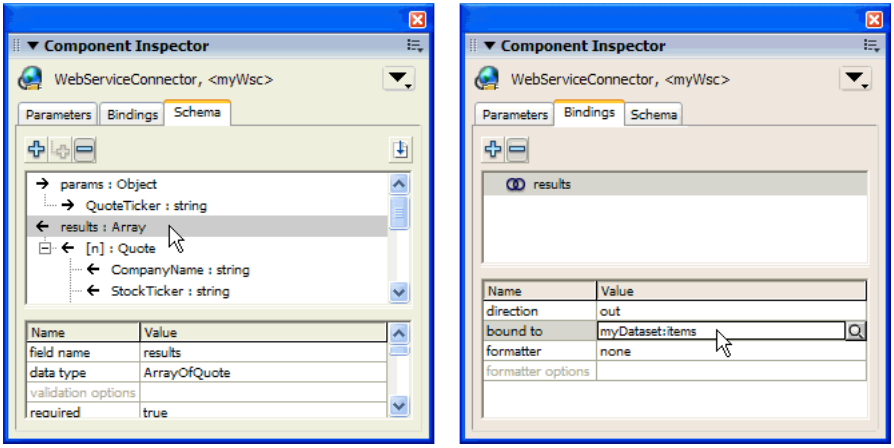
```
function loadData() {
   var recData = new Array();
   for( var i:Number=0; i<100; i++ ) {
      recData[i]= {id:i, name:String("name"+i), price:i*.5};
   }
```

```
    myDataSet.items = recData;
}
```

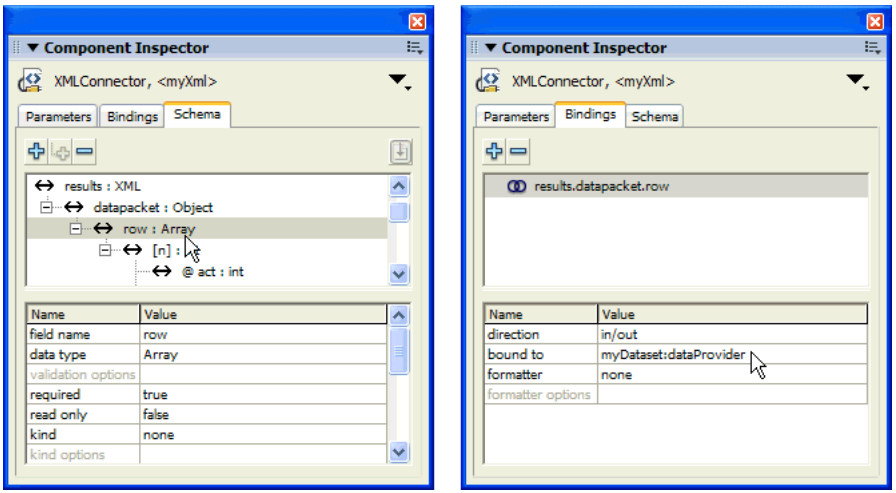
Remoting RecordSet The following ActionScript example assumes that you're using Flash Remoting and that you've made a remoting call that returns a RecordSet. The RecordSet object implements the DataProvider interface. The result is assigned to the `dataProvider` property of the `myDataset` component instance:

```
function getSQLData_Result(result) {
  myDataset.dataProvider = result;
}
```

Array of objects returned from a web service The following illustration shows an example of using the Component inspector to bind an array of objects returned from the web service, represented by the `myWsc` instance of the WebServiceConnector component. The illustration on the left shows the schema of the web service. The illustration on the right shows how the `results` array is bound to the `items` property of the `myDataset` component instance.



Array of objects returned from an XMLConnector component The following illustration shows an example of using the Component inspector to bind an array of XML nodes, represented with the XMLConnector component. It assumes that you have imported a schema for an XML file that contains an array of XML nodes. The illustration on the left shows the schema of the XML document, the array of XML nodes represented as an ActionScript array. The illustration on the right shows how the `results.datapacket.row` array is bound to the `dataProvider` property of the `myDataset` instance of the DataSet component.



Accessing the data

After the data is loaded into the DataSet component and the schema for the DataSet component has been defined, the data can be accessed. You can access data at runtime or at design time.

Runtime example. Accessing the data at runtime is simple. Because the data is loaded as objects, data is exposed through properties that can be referenced in code. The DataSet component has a method (`DataSet.first`) that lets you make the first item in the array the currently selected object.

The following code shows an example of accessing data at runtime. It loads an existing DataSet component instance `myDataSet` with customer information and then displays each customer's name in the trace window. Notice that the data types for the customer information—the array of objects—are added so the data displays properly:

```
//Drag DataSet component to Stage and name it myDataSet (easiest way to create
  instance and import necessary libraries)

//Creates recData which contains customer information in an array of objects
var recData = [{id:0, firstName:"Frank", lastName:"Jones", age:27,
  usCitizen:true},
            {id:1, firstName:"Susan", lastName:"Meth", age:55,
  usCitizen:true},
            {id:2, firstName:"Pablo", lastName:"Picasso", age:108,
  usCitizen:false}];
```

```
//Assigns recData to the items property of the “myDataSet” DataSet component
  instance
myDataSet.items = recData;

//Adds schema types for the expected fields
var i:mx.data.types.Str;
var j:mx.data.types.Num;

//Makes the first item the current item
myDataSet.first();

//Traces through the properties
while ( myDataSet.hasNext() ) {
  //access the data through the Dataset properties
  trace(myDataSet.firstName + " " + myDataSet.lastName);
  myDataSet.next();
}
```

Design time example. Creating fields for a DataSet component at design time is another way to expose the properties of a data object. After the fields are defined, you visually bind UI controls to the data at design time. You can set additional properties (schema item settings) at design time for a DataSet field to affect the way data is encoded, formatted, and validated at runtime. For more information, see “Schema item settings” on page 289.

To set up binding to this data at design time, you create persistent fields for the DataSet component that represent the properties of the object. The following procedure shows an example of how you would access the same customer information data at design time. You bind the `recData` array of objects to the items property of the DataSet component in ActionScript, as in the runtime example. Then, you bind `DataGrid.dataProvider` into `myDataSet.items` using the Component inspector.

To access data at design time:

1. Drag a DataSet component onto the Stage. Name it **myDataSet**.
2. Select a layer in the Timeline, and press F9 to open the Actions panel. Type the following code:

```
var recData = [{id:0, firstName:"Frank", lastName:"Jones", age:27,
  usCitizen:true},
            {id:1, firstName:"Susan", lastName:"Meth", age:55,
  usCitizen:true},
            {id:2, firstName:"Pablo", lastName:"Picasso", age:108,
  usCitizen:false}];
myDataSet.items = recData;
```

3. With the DataSet component selected, click the Schema tab of the Component inspector, and click the Add a Component Property (+) button.
4. Set the value for Field Name to **firstName** and leave the Data Type as String.
5. Create three more component properties for the other name/value pairs in the code: field name = lastName, data type = String; field name = usCitizen, data type = Boolean; and field name = age, data type = Integer.
6. Drag a DataGrid component onto the Stage, and name it **myGrid**.

7. Select the DataGrid component, and click the Bindings tab of the Component inspector.
8. Click the Add Binding (+) button to add a new binding. Select `dataProvider:Array`.
9. Click Bound To, select the DataSet component, and select its `dataProvider:Array` property.
10. Click Direction and select In.
11. Save and test the application.

    The data contained within the data set appears in the data grid.

The ability to make use of dynamic component properties that are added to the Schema tab at design time is a special feature of the DataSet component. The DataSet component uses the field name of these properties to map them to the properties of the object or array of objects. The settings that are applied to these properties at design time are then used by the data set at runtime.

If you do not create persistent fields for the DataSet component and you bind it to a WebServiceConnector component or an XMLConnector component that defines a schema, the DataSet component tries to create the correct fields based on the connector component's schema, which might not work. For more information, see "Managing data with the DataSet component (Flash Professional only)" on page 281.

Note: Persistent fields that are defined for a DataSet component take precedence over the schema for a connector component.

Data resolution (Flash Professional only)

The resolver components let you convert changes made to the data within your application into a format that is appropriate for the external data source that you are updating. The resolver components can also receive updates from an external data source and convert them into a format that is appropriate for the DataSet component to receive them.

Flash MX Professional 2004 includes the following resolver components:

- "XUpdateResolver component (Flash Professional only)" for XML data sources
- "RDBMSResolver component (Flash Professional only)" for relational databases

Typically, you use the resolver components with the DataSet component. When a user edits data in your application, the data is captured in the DataSet component. The DataSet component generates a DeltaPacket, an object that contains a list of changes made to the data at runtime. The resolver component then converts the DeltaPacket to the appropriate format (update packet). When an update is sent to the server, the server should respond with a results packet containing errors or updated field values from the operations that were performed. The resolver components can convert this information back into a DeltaPacket that can then be applied to the data set to keep it synchronized with the external data source.

Tip: The RDBMSResolver component provides limited synching ability at this time.

Resolver components do not send any data from a SWF to server-side scripts or external data sources. You need to set up this kind of data transfer. Here are the most common ways to send data outside a SWF:

- Bind the resolver's processed data to a connector component, such as the XMLConnector or WebServiceConnector components. This connector component instance is in addition to the instance that connects your data source to a DataSet or to UI components; see the diagram at the beginning of this chapter.
- Write ActionScript code using the LoadVars class (see "LoadVars class" in *Flash ActionScript Language Reference)*.
- Write ActionScript code using the XML class (see "XML class" in *Flash ActionScript Language Reference*).

For more information, see Chapter 11, "Working with External Data" in *Flash ActionScript Language Reference*.

Note: External data refers to any data that is accessible through HTTP.

Resolving XML data with the XUpdateResolver component (Flash Professional only)

The XUpdateResolver component converts changes made to the data in your application into XUpdate statements that can be processed by an external data source. XUpdate is a standard for describing changes that are made to an XML document and is supported by a variety of XML databases, such as Xindice and XHive. You can write your own server code to handle updates, for example, in your own ASP page, Java servlet, or ColdFusion component. For more information, see the XUpdate specification at www.xmldb.org/xupdate/.

The XUpdateResolver component works only in applications published for Flash Player 7.

For a common workflow and information about the methods, events, and properties of the XUpdateResolver component, see "XUpdateResolver component (Flash Professional only)" in *Using Components*.

You need to set the correct encoder when you use the XUpdateResolver component; for more information, see the discussion of the DatasetDeltaToXUpdateDelta encoder in "Schema encoders" on page 294.

For a tutorial that uses this component, see the XUpdate tutorial, "XML Tutorial: Timesheet (Flash Professional Only)," in the Data Tutorials in Flash Help.

Updates sent to an external data source

When a user edits data in your Flash application, the data is captured in the DataSet component. The DataSet component produces a DeltaPacket, which the resolver component uses to create an update packet. The update packet consists of XUpdate statements, which are communicated to an external data source through a connector component. These statements describe the inserts, edits, and deletes performed on the DataSet component. You can view or bind the contents of the update packet using the `xupdatePacket` property of the XUpdateResolver component.

Note: The information contained within the XML update packet is affected in part by the component parameter values that are assigned by the developer. For information on the XUpdateResolver component parameters, see "Using the XUpdateResolver component (Flash Professional only)" in *Using Components*.

The following XML code is an example of an update packet created by an XUpdateResolver component:

```
<?xml version="1.0"?>
<xupdate:modifications version="1.0" xmlns:xupdate="http://www.xmldb.org/
  xupdate">
   <xupdate:insert-after select="/addresses/address[1]" >
      <xupdate:element name="address">
         <xupdate:attribute name="id">2</xupdate:attribute>
         <fullname>Lars Martin</fullname>
         <born day='2' month='12' year='1974'/>
         <town>Leizig</town>
         <country>Germany</country>
      </xupdate:element>
   </xupdate:insert-after>
</xupdate:modifications>
```

When you use the XUpdateResolver component with a DataSet, you must set the correct encoder on the Schema tab: the DataSetDeltaToXUpdateDelta encoder. This encoder is responsible for creating XPath statements that uniquely identify nodes within an XML file based on the information contained within the DataSet component's DeltaPacket. This information is used by the XUpdateResolver component to generate XUpdate statements. For more information about the DataSetDeltaToXUpdateDelta encoder, see "Schema encoders" on page 294.

In addition to client-side code and configuration, you or your server administrator also need to write server code to handle the interaction with your Flash application. For more information, see "Server-side requirements for resolving XML data" on page 303.

Resolving data to a relational database (Flash Professional only)

The RDBMSResolver component creates an XML packet that can be sent to an external data source (such as ASP/JSP page, servlet, and so on). The XML packet can easily be translated into SQL statements that can be used to update any standard SQL relational database. Your development team must write the server code to parse the XML and generate SQL statements.

You can use the RDBMSResolver component to send data updates to any external data source that can parse XML and generate SQL statements against a database—for example, an ASP page, a Java servlet, or a ColdFusion component.

When an RDBMSResolver component receives a delta packet from a DataSet component, it converts it into an XML update packet, which can be communicated to an external data source through a connector component. The converted output is referred to as an update packet and consists of an optimized set of instructions that describe the inserts, edits, and deletes performed on the DataSet component. You can view or bind the contents of the update packet using the `xupdatePacket` property of the RDBMSResolver component.

The RDBMSResolver component works only with Flash Player 7.

For a typical workflow and information on the methods, properties, and events of the RDBMSResolver class, see "Using the RDBMSResolver component (Flash Professional only)" and "RDBMSResolver class (Flash Professional only)" in *Using Components*.

In addition to requirements for your Flash application to resolve data, there are requirements for your server code to fulfill. For more information, see "Server-side requirements for resolving data for RDBMS" on page 304.

For a tutorial that uses the RDBMSResolver component, see the DevNet article "Using the RDBMSResolver to Update a Database" at www.macromedia.com/devnet/mx/flash/articles/delta_packet.html.

Formatting your results (Flash Professional only)

By default, the resolver components use the schema specified on the connector components to format values sent to the server. This method ensures that a date value sent from an external data source using the format "MM/DD/YYYY" is sent back to the external data source using the same format.

However, in some cases, you might find that the values you're sending to your external data source are not formatted correctly. This can occur when you don't use a connector to retrieve your data or you want to change the format of the data to be sent to an external data source. In this case, you can control the formatting by adding properties to the resolver component's schema. For instance, if you have a Boolean field called Billable in your DataSet component, its value can be formatted in an update packet as `true` or `false`. If you want it formatted as `yes` or `no`, you can create a new component property called `Billable` within the Schema tab for your resolver. Using the schema settings, you can set the data type as Boolean, the encoder as Boolean, and the encoder options as `yes` or `no`. The encoder is applied when the resolver creates the update packet, and the value for the billable field is represented as `yes` or `no`.

For more information, see "Adding a component property to a schema" on page 262.

Advanced topics in data integration

This section discusses advanced topics, such as refinements you make to schema settings and information for developers who need to write server-side code to interact with Flash data applications.

Schema item settings

This section contains details about schema item settings and how you edit them. To help you determine whether or not you need to look at schema item settings, see "When to edit schema item settings" on page 300.

The schema of a component shows what properties and fields are available for data binding. For each property or field, there are settings that control validation, formatting, type conversion, and other features that affect how data binding and the data management components handle the data of a field. The Schema Attributes pane, the bottom pane of the Schema tab, presents these settings, which you can view and edit. The following list describes the five categories of settings, according to the features they control:

Basic settings Every field or property has these basic schema settings. In many cases, these are the only settings you need to bind to a field:

- Name: Every field needs a name.
- Data type: Every field has a data type, which is selected from a list of available data types. The data type of a field affects data binding in two ways: When a new value is assigned to a field through data binding, the data type determines the rules that are used to check the data for validity. When you bind between fields that have different data types, the data binding feature attempts to convert the data appropriately. For more information, see "Schema data types" on page 298.
- Storage type: Every field has a storage type. Typically, it defaults to one of four values based on the data type of a field. The available values for storage types are simple, attribute, array, or complex.

  Note: Developers almost never have to change this setting. However, there are some cases when the storage type for an attribute contained within the schema for an XML file should be set to attribute.

- Path (optional): This property identifies the location of the data for this schema field. For more information, see "Virtual schemas" on page 301 and "Setting the schema path" on page 293.

Validation settings Validation settings are applicable to any field that is the destination of a binding. You usually modify these settings when you want to control the data validation that the user inputs. To do so, you bind from the UI component to a data component, and then select appropriate validation settings for the fields of the data component. One common example is when the user input is bound to the `params` property of a connector component, such as the XMLConnector component or WebServiceConnector component. Another common example is when UI components are bound to data fields of the DataSet component.

This is how validation works: After any binding is executed, the new data is checked according to the validation rules of the destination field's data type. A component event is then generated to signal the results of the checking. If the data is valid, then the valid event is generated; otherwise, an invalid event is generated. Both components involved in the binding emit the event. You can ignore these events. If you want anything to happen as a result of these events (such as giving feedback to the user), you must write some ActionScript code that receives the valid and/or invalid events.

- Validation Options: Validation options are additional settings that affect the validation rules for this field. The settings are presented in the Validation Options dialog box, which appears when you select this item. These settings vary according to data type. For example, the String data type has settings for the minimum and maximum allowed length of the data. The XML data type has a setting to control if white space is ignored when converting from a String to XML.
- Required: This is a Boolean value that determines whether this field is required to have a non-null value. Validation fails if `required=true` but no value has been set.
- Read-Only: This is a Boolean value that determines whether this field can receive new values through data binding. If `readonly=true`, then executing any binding to this field generates the invalid event, and the field changes.

Formatter settings Formatter settings are applied when a field's value needs to be converted to a string. This is often for display purposes, such as when a DataSet field is bound to the `text` property of a Label or TextArea component. Formatter settings on a field are ignored when that field is bound to something whose data type isn't String.

- Formatter: The name of the formatter to use when converting this field to String. This is selected from a list of available formatters.
- Formatter options: these additional settings affect the formatter. The settings are presented in the Formatting Options dialog box, which appears when you select this item. These settings vary according to formatter. For example, the Boolean formatter has settings for the text that represents the `true` and `false` values.

Note: If you don't specify a formatter, then a default conversion is applied when a field's value is needed as a string.

For a complete list of formatters, see "Schema formatters" on page 297.

Kind and Encoder settings The Kind and Encoder settings are used to activate certain special features:

- Kind: The Kind setting for this field. This is selected from a list of available Kind settings.
- Kind options: Additional settings that affect the Kind setting. The settings are presented in the Kind Options dialog box, which appears when you select this item. These settings vary according to kind.
- Encoder: The Encoder setting for this field, which is selected from a list of available Encoder settings.
- Encoder options: Additional settings that affect the encoder. The settings are presented in the Encoder Options dialog box, which appears when you select this item. These settings vary according to encoder.

For more information, see "Using kinds and encoders" on page 292, "Schema kinds" on page 294, and "Schema encoders" on page 294.

Default settings These settings let you set defaults for various situations. The following list describes the uses for these settings:

- If a field's value is undefined, the default value is used whenever the value of the field is used as the source of a data binding. For example, the data fields of a DataSet component, or the `results` property of a connector component, can have an undefined value.
- When you create a new row of data in a DataSet component, the default value is used as the value of newly created records

Using kinds and encoders

Kinds and encoders are drop-in modules that perform additional special processing of the data of a schema item. They are often used with each other to accomplish common tasks. The following list describes common uses for kinds and encoders:

Calculated DataSet Fields Calculated fields are virtual fields that do not exist in the underlying data tables. Calculated fields provide developers with the ability to create and update dynamic field values at runtime. This is convenient for calculating and displaying values based on calculations or concatenations performed other fields located in a record (for instance, you can create a calculated field that combines the first and last name fields together to display the full name to a user).

To set up a calculated field for the DataSet component:

1. Select the DataSet component, and click the Schema tab in the Component inspector.
2. Click the Add a Component Property (+) button. This step adds a field to the schema.
3. Using the Schema Attributes pane, give the new component property a field name, and set its kind to `calculated`.
4. In ActionScript code, use the `calcFields` event of the DataSet component to assign this field a value at runtime.

   Note: You should assign a value to a calculated field only within the DataSet component's `calcFields` event.

For an ActionScript code example, see "Schema kinds" on page 294.

Setting up schemas for XML documents In an XML document, all data is stored as a string. Sometimes you want the fields of an XML document to be available as data types other than String. The following example shows an application that pulls in data from an XML file:

```
<datapacket>
   <row id="1" billable="ON" rate="50" hours="3" />
   <row id="2" billable="OFF" rate="50" hours="6" />
</datapacket>
```

If you use this XML file to import a schema for the XMLConnector component's `results` property, it generates the following code:

```
results : XML
   datapacket : Object
      row : Array
         [n] : object
```

```
@billable: String
@hours : Integer
@id : Integer
@rate : Integer
```

Suppose you want to treat the row node as a record within a grid, and you want the `@billable` attribute to be treated as a Boolean value and show a `true` or `false` value in the grid instead of `ON` or `OFF`. Getting the data into the grid is simple: You can simply bind the row schema field to the `dataProvider` property of the grid. The following procedure describes how to get the `@billable` attribute to be treated as a Boolean value and display a `true` or `false` value.

To make the @billable attribute display a true or false value:

1. Select the XMLConnector component, click the Schema tab, and select the @billable schema field.
2. In the bottom pane of the Schema tab, set the `data type` property to Boolean.
3. Set the `encoder` property to Boolean.
4. Select Encoder Options and enter **on** for strings that represent `true`, and enter **off** for strings that represent `false`.

   The encoder now takes the XML data in its raw form (String) and converts it into an ActionScript Boolean value. Using the encoder options, it knows how to encode the string values correctly.
5. Click Formatter, and select Boolean. Select Formatter Options. You now have a choice to define how a `true` and `false` value should appear as a string.
6. Enter **True** for strings that mean `true`, and enter **False** for strings that mean `false`.

   The formatter now takes the ActionScript Boolean value and formats it into a String.

Setting the schema path

The `path` property for a schema field is an optional setting that is used in special circumstances when the schema for your component is not appropriate. Using this setting, you can create a virtual schema field (a field that exists in one location but pulls its value from another). The value of this property is a path expression that is entered in one of the following formats:

- For schemas that contain ActionScript data, the path follows the format `field [.field]...`, where `field` is equal to the name of a field (such as `addresslist.street`).
- For schemas that contain XML data, the path follows the format `XPath`, where `XPath` is a standard XPath statement (such as `addressList/street`).

When data binding is performed, Flash checks to see if there is a path expression for a schema field. If so, it uses the path expression to locate the correct value. For more information, see "Virtual schemas" on page 301.

Note: The path expression is always performed relative to the parent node of the schema field.

Schema kinds

A kind determines how a schema item for your component should be accessed at runtime. The following kinds come with Flash MX Professional 2004:

None The default kind. This kind is identical to the `Data` kind.

Data The schema item is a data structure, and the data field is stored within the data structure as specified by the field's schema location. This is the normal case. The data structure can be in either ActionScript or XML form.

Calculated This kind is used with the DataSet component. You can use it to define a calculated field (a virtual field whose value is calculated at runtime, based on the values of other fields). You write an event handler in ActionScript code that is invoked by the `DataSet.calcFields` event when any non-calculated field in a data set's current data record changes. The event handler must set the value of the calculated fields in that record. There is no special processing when getting or setting the value of a calculated field. For example, in the DataSet component you might define three fields, called `price`, `quantity`, and `totalPrice`. You would set the `kind` property for `totalPrice` to `Calculated` so that you can assign it a value at runtime, as shown in the following example:

```
function calculatedFunct(evt) {
      evt.target.totalPrice = (evt.target.price * evt.target.quantity);
    }
ds.addEventListener('calcFields', calculatedFunct);
}
```

See the `DataSet.calcFields` event in *Using Components*.

AutoTrigger This kind can be applied to any property of any component but is mainly useful for connector component properties. When a new value is assigned to the property through data binding, the trigger method of the component is called. For more information, see `WebServiceConnector.trigger()` and `XMLConnector.trigger()` in *Using Components*.

You can create custom kinds. The number of kinds allowed is unlimited. Kinds are defined by XML files found in the Flash MX Professional 2004 Configuration/Kinds folder. The definition includes the following metadata:

- An ActionScript class that will be instantiated to mediate access to the data
- A Kind Options dialog box

Schema encoders

An encoder determines how a schema item for your component should be encoded/decoded at runtime. Sometimes you might want a component property to have a different data type that what is actually stored inside the component. For example, an XMLConnector component results property is stored as an XML document, which contains only strings. You might want a certain field in the results to appear as a Boolean value instead.

To do this, you set the field's data type to Boolean, which tells the data binding mechanism to expect Boolean values in that field; and you set the field's encoder to Boolean, which performs the translation between the underlying string value and the Boolean value that data binding expects the property to have. See the example in "Using kinds and encoders" on page 292.

The following encoders come with Flash MX Professional 2004:

None The default encoder. No encoding/decoding occurs.

Boolean Converts data of the String type into the ActionScript Boolean type. You must specify (using the Encoder Options property) one or more strings, separated by commas, that will be interpreted as `true`, and one or more strings that will be interpreted as `false`. The settings are case-sensitive.

Date Converts data of the String type into the ActionScript Date type. You must specify (using the Encoder Options property) a template string, which works as follows:

- The template string should contain 0 or 1 instances of `"YYYY"`, `"MM"`, `"DD"`, `"HH"`, `"NN"`, and/or `"SS"`, mixed with any other combination of characters.
- When converting from date to string, the numeric year, month, date, hour, minutes, and seconds, respectively, are substituted into the template, in place of YYYY, MM, and so on.
- When converting from string to date, the string must *exactly* match the template, with the correct number of digits for each of year, month, day, and so on.

DateToNumber Converts a Date object into its numeric equivalent. The DataSet component uses this encoder for fields that are of the Date type. These values are stored within the DataSet component as numbers so that they can be sorted correctly.

Number Converts data of the String type into the ActionScript Number type. There are no authoring settings for this encoder.

DatasetDeltaToXUpdateDelta You use this encoder extracts information from a DeltaPacket and generates XPath statements that are passed to the XUpdateResolver component to generate XUpdate statements. It gets the information that is needs to generate the XPath statements from two places:

- The `rowNodeKey` property, which you must specify with the Encoder Options property (defined in the third bullet, below).
- Within the schema that was used for the XMLConnector component that originally retrieved the data.

Using this information, the encoder can generate the correct XPath statements needed to identify your data within the XML file.

The encoder options contain one property:

- The `rowNodeKey` property (String type). In order for an XML file to be updated, the file must be structured in such a way that the node that represents a record in your data set can be uniquely identified with an XPath statement. This property combines an XPath statement with a field parameter to uniquely identify the row node within the XML file and the field within the data set that makes it unique.

In the following example, the row node represents a record within the XML file. The value of the `id` attribute is what makes the row unique.

```
<datapacket>
   <row id="1" date="01/01/2003" rate="50" hours="5" />
   <row id="2" date="02/04/2003" rate="50" hours="8" />
</datapacket>
```

The XPath to uniquely identify the row node is shown in the following example:

```
datapacket/row[@id='xxx']
```

In this example, xxx represents a value for the `id` attribute. In a typical case, the `id` attribute in the XML file would be bound to the `id` field of the DataSet component. Therefore, the `rowNodeKey` value would be as follows:

```
datapacket/row[@id='?id']
```

The question mark symbol (`?`) identifies that this is a field parameter. The `id` value specifies the name of the field in the data set. At runtime, the XUpdateResolver component substitutes the value from the `id` field of the data set to generate the correct XPath for the specified record.

In the next example, the contacts node with a category attribute of Management represents the record(s) within the XML file, and the `employeeId` subnode contains the value that makes the record unique:

```
<datapacket>
   <company id="5" name="ABC tech">
      <contacts category="Mgmt">
        <contact>
          <empId>555</employeeId>
          <name>Steve Woo</name>
          <email>steve.woo@abctech.com</email>
        </contact>
        <contact>
          <empId>382</employeeId>
          <name>John Phillips</name>
          <email>john.phillips@abctech.com</email>
        </contact>
        ...
        ...
      </contacts>
      <contacts category="Executives">
        ...
        ...
      </contacts>
      ...
      ...
   </company>
</datapacket>
```

The `rowNodeKey` value for this XML file would be as follows:

```
datapacket/company/contacts[@category='Mgmt']/contact[empId='?empId']
```

You can create custom encoders. The number of encoders allowed is unlimited. Encoders are defined by XML files found in the Flash MX Professional 2004 Configuration/Encoders folder. The definition includes the following metadata:

- An ActionScript class that will be instantiated to encode/decode the data. This class must be a subclass of mx.databinding.DataAccessor.
- An Encoder Options dialog box

Schema formatters

A formatter is an object that performs bidirectional conversion of data between a raw data type and string data. The object has parameters that are settable during authoring and runtime methods for performing the conversion. The following formatters come with Flash MX Professional 2004:

None The default formatter. No formatting is performed.

Boolean This formatter formats a Boolean value as a string. You can set up Boolean options for strings that mean `true` (for example, 1, yes) and strings that mean `false` (for example, 0, no).

Compose String This formatter converts a data object to a string. You define the output format using a string template. The template is arbitrary text that can refer to the data fields in one of the following ways:

- `<field-name>`
- `<field-name.field-name>`, using dots to drill down into the data structure
- `<.>`, which represents the entire object. This can be used, for example, when the original object is a string, in which case `<.>` is simply the value of the string.

Here are two examples using the Compose String formatter. A formatter could be applied to a field that is an object with field name, quantity, and price, and the string output could read: "`You ordered <quantity> units of <name> at <$price> each`." In another example, the formatter could be applied to a field that is a number, and you could define the string output to read: "`You have <.> messages.`"

Custom Formatter This formatter lets you specify a custom formatter by specifying a class name. The formatter ActionScript class should have the following format:

```
class MyFormatter extends mx.data.binding.CustomFormatter {
  // convert a raw value, returns a formatted value
  function format(rawValue){
  }
  // convert a formatted value, returns a raw value
  function unformat(formattedValue){
  }
}
```

Rearrange Fields This formatter creates a new array of objects based on the original array in your binding. It can only be applied to fields that are arrays. You define the fields on the new array by using a string template in the form:

```
fieldname1=definition1;fieldname2=definition2;and so on.
```

The `fieldnameN` are the names of the fields in the new array or records. The `definitionN` is one of the following:

- The name of a field in the original record
- A string, enclosed in single quotes ('), that contains a mix of text and tags. A tag is the name of a field in the original array, enclosed in < and >.
- A single dot (.), which represents the entire original record

For example, suppose you want to assign an array to the `DataProvider` property of a List component using data binding. The objects within the array do not have a `label` property (which the list uses if available). You could use this formatter to create a new array through data binding that replicates the objects within your original array and adds a `label` property to each object using the values you define. The following template would achieve this (this would be on a binding between your array and the List component's `DataProvider` property):

```
label='My name is <firstName> <lastName>;'
firstName=firstName;
lastName=lastName;
```

This syntax assumes that the object has two properties, called `firstName` and `lastName`. The `label` property will be added to each object within the new array.

Note: This formatter can be used on any binding from a component property that is of the Array type to another component property of the Array type. Also note that the Rearrange Fields formatter doesn't work if you access if in the Schema panel, but does work if you access it in the Bindings panel.

Number Formatter This formatter allows you to specify the number of fractional digits that appears when a number is converted to text.

You can create custom formatters. The number of formatters allowed is unlimited. Formatters are defined by XML files found in the Flash MX Professional 2004 Configuration/Formatters folder. The definition includes the following metadata:

- The ActionScript class that will be instantiated to perform the formatting
- A Formatter Options dialog box

Schema data types

A data type is an object that represents all the runtime logic needed to support a particular data type. A data type can be a scalar type, such as integer, string, date, currency amount, or ZIP code. It can also be a complex type, with subfields and so on. A data type can test a data value to determine if it is valid for that data type. The following data types come with Flash MX Professional 2004:

Array No validation options.

Attribute XML attribute. No validation options.

Boolean No validation options.

Custom Lets you add a custom class to check for this special kind of validation. Your code should call the validate function when the field is assigned a new value, inspect the value, and determine whether it's valid. If it is, the function should simply return. If not, the function should call `this.ValidationError("some informative message");`. The custom class must be in the classpath and formatted as shown in the following example:

```
class myCustomType extends mx.databinding.CustomValidator {
  function validate(value) {
      ... some code here
   }
}
```

DataProvider No validation options.

Date No validation options.

DeltaPacket No validation options.

Integer A validation option can be set up to define the minimum and maximum values.

Number A validation option can be set up to define the minimum and maximum values.

Object No validation options.

PhoneNumber No validation options.

SocialSecurity No validation options.

String A validation option can be set up to define the minimum and maximum number of characters.

XML Lets you specify if white space should be ignored when a string is converted into XML.

ZipCode No validation options.

Note: The following data types can perform validation: Custom, Integer, Number, PhoneNumber, SocialSecurity, String, ZipCode. The following data types can convert from various other data types when you assign to them: Boolean, DataProvider, Integer, Number, String, XML.

You can create custom data types. The number of data types allowed is unlimited. Data types are defined by XML files found in the Flash MX Professional 2004 Configuration/DataTypes folder. The definition includes the following metadata:

- An ActionScript class that will be instantiated for validation and type conversion
- A Validation Options dialog box
- The name of the standard formatter, which you can override using the `formatter` property
- Initial values for required, read-only, and default values

When to edit schema item settings

You can edit anything within the Schema Attributes pane, even schemas that come from an external source, such a web service WSDL file. You can always change any value for any field of any schema, with the following restrictions:

- If you change the type, all the other schema item attributes are reset to the default values for the new data type.
- If you select to completely reload the schema for a component property, you will lose all the edits that had previously been made within the Schema Attributes pane.

  Note: There are several ways to reload the schema for a component property, including entering a new WSDL URL, selecting a different operation for a web service, or importing a new XML schema from a sample XML file.

When you build an application using data components and/or data binding, you need to apply schema item settings to some, but not necessarily all, fields of the components in your application. The following table summarizes the most common uses of schema item settings and can help you determine when these settings need to be edited:

Component	Property/field	Settings	When to use
Any connector	`params` (and its sub-fields)	Validation Options, Read-Only, Required	If validation is desired.
	`results` (and its sub-fields)	Formatter, Formatter Options	For fields that need formatting for display as text.
		Default value	For fields whose value is sometimes undefined.
DataSet	Any data field	Name, Data Type	You must set these for every data set field that you define.
		Validation Options, Read-Only, Required	If validation is desired.
		Formatter, Formatter Options	For fields that need formatting for display as text.
		Default Value	For fields whose value is sometimes undefined, or to specify the initial value for newly created data set records.
UI components	UI components typically don't need any changes to their schema settings.		

Component	Property/field	Settings	When to use
Any component	Any property or field	Kind, Kind Options, Encoding, Encoding Options	Various purposes, as described in "Using kinds and encoders" on page 292.
Any connector	`results` (and its subfields)	Path	To identify the location of the data for a virtual schema field.

Virtual schemas

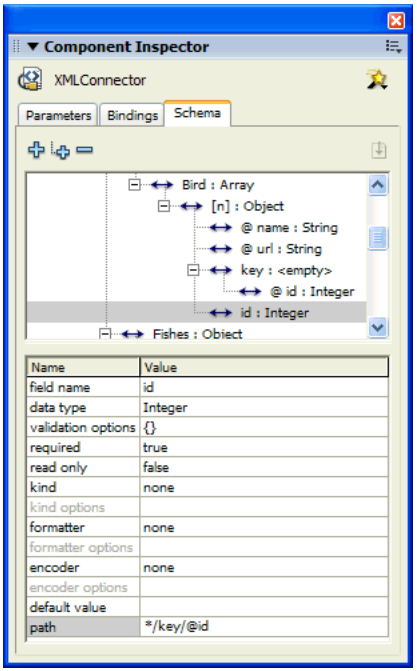
When you bind an array of data to a DataSet's `items` or `dataprovider` property, the data set only recognizes fields that are top-level items within each row of the array. It does not recognize items nested within other objects. A virtual schema lets you change how the underlying data structure is interpreted when bindings are executed. The new structure is derived using XPath statements. For more information, see "Adding bindings using path expressions" on page 302.

For example, the schema for Animals.xml file described in "Connecting to XML data with the XMLConnector component (Flash Professional only)" on page 277 defines an array of objects called Bird. Each object contains two fields (`name` and `url`). They also contain a sub-element with one field called `id`. If you bind the Bird array to a DataSet component (using the `dataProvider` property) with three fields—`name`, `url`, and `id`—each item that is returned from the array is constructed in the following way, for each item in the XML file:

- Create an empty item.
- Loop through the defined schema properties, extracting the values for each property from the XML data, and assign these values to the created item. The Name and URL fields will have values.
- Provide this item to the DataSet component.

  The ID field does not exist on the item, and the DataSet component has a blank entry for each item assigned.

The solution is to create a new schema field under the object within the Bird array. The new schema field is named `id`. Every schema field has a property called `path` that accepts an XPath statement that points to the data in your XML file. In this case the XPath expression would be `key/@id`. When you get to the second bullet in the above process, data binding finds an `id` field for the object. It looks at the `path` property and uses the XPath statement to get the correct data from the XML file. The correct data is then passed to the DataSet component.



Adding bindings using path expressions

You can use path expressions for data binding in two areas:

- In the Add Binding dialog, to identify the field you are binding to
- In the Bound To dialog box, to identify the field you're binding from.

The following XPath expressions are supported:

- Absolute paths:
  `/A/B/C`
- Relative paths:
  `A/B/C`
- Node selection using node name or wildcard:
  `/A/B/C` (node selection by name)
  `/A/B/*` (node selection of all child nodes of /A/B by wildcard)
  `/*/*/C` (node selection of all C nodes that have exactly two ancestors)

- Predicate syntax to further specify nodes to be selected:

  `/B[C]` (child node syntax; selects all B nodes that have a C node as a child)

  `/B[@id]` (attribute existence syntax; selects all B nodes that have an attribute named `id`)

  `/B[@id="A1"]` (attribute value syntax; selects all B nodes that have an `id` attribute whose value is A1)
- Support for predicate comparison operators:

  =
- Support for Boolean `AND` and `OR` values in predicates:

  `/B[@id=1 AND @customer="macromedia"]`

Note: The following operators are not supported: "<", ">", "//".

To add a binding using path expressions:

1. In either the Add Binding dialog box or the Bound To dialog box, select Use path expression.
2. Enter a path expression to identify the schema item to which you want to bind. Path expressions are entered in the following formats:
   - For properties that contain ActionScript data, the path follows this format:

     `field [.field]...`

     In this format, `field` is equal to the name of a field (such as `addresslist.street`).
   - For properties that contain XML data, the path follows this format:

     `XPath`

     In this format, `XPath` is a standard XPath statement (such as `addressList/street`).
3. Click OK to return to the Bindings tab.

Default data binding events

When you use the Bindings tab to create a binding between two components, the binding is triggered by the default component event. If you want a binding to execute independently of the default component event (which is predetermined by Flash), you must manually refresh the binding with ActionScript code. For more information, see "ComponentMixins class (Flash Professional only)" in *Using Components* (in particular, see the `ComponentMixins.refreshDestinations()` and `ComponentMixins.refreshFromSources()` methods).

In general, for UI components, the `change` or `click` events are the default events used to trigger data bindings, such as `TextInput.change`, `Button.click`, `RadioButton.click`. For connector components, the `result` event triggers the binding, such as `XMLConnector.result`.

Server-side requirements for resolving XML data

This section describes requirements that your server code must fulfill when receiving results from an XUpdateResolver component. It contains information relevant for the server administrator who is handling server-side functions for your Flash application.

After the server finishes with the update packet, either successfully or unsuccessfully, it should send back to your Flash application a results packet containing errors or additional XML updates resulting from the update operation. If there are no messages, the results packet should still be sent, but it will have no operation result nodes.

The following example shows a sample results packet for an update packet that has no errors and contains no XML updates:

```
<results_packet nullValue="{_NULL_}" transID="46386292065:Wed Jun 25 15:52:34
  GMT-0700 2003"/>
```

A sample results packet (with XML updates) follows:

```
<results_packet nullValue="{_NULL_}" transID="46386292065:Wed Jun 25 15:52:34
  GMT-0700 2003">
    <operation op="remove" id="11295627479" msg="The record could not be
  found"/>
    <operation op="update" id="02938027477">
       <attribute name="id" curValue="105" msg="Invalid field value" />
    </operation>
 </results_packet>
```

The results packet can contain an unlimited number of operation nodes. Operation nodes contain the results of operations from the update packet. Each operation node should have the following attributes/child nodes:

- `op`: An attribute describing the type of operation that was attempted. Must be insert, delete, or update.
- `id`: An attribute that holds the ID from the operation node that was sent out
- msg (optional): An attribute containing a message string that describes the problem that occurred when attempting the operation
- `field`: 0, 1, or more child nodes that give field-level specific information. Each field node, at a minimum, should have a `name` attribute, which contains the field name, and a `msg` attribute, which gives the field-level message. It can also optionally contain a `curValue` attribute, which holds the most up-to-date value for that field in that row on the server.

Server-side requirements for resolving data for RDBMS

This section describes requirements that your server code must fulfill. It contains information relevant for the server administrator who is handling server-side functions for your Flash application. It contains the following topics:

- Example of an RDMBSResolver component XML update packet
- About receiving results from an external data source

In addition to the information in this section, see the DevNet article "Using the RDBMSResolver to Update a Database" at www.macromedia.com/devnet/mx/flash/articles/delta_packet.html.

Example of an RDMBSResolver component XML update packet

To handle server-side code, you'll need to understand the XML update packet generated by the resolver component. The information contained within the XML update packet is affected in part by the component parameter values that are assigned by the developer. For information on the RDBMSResolver component parameters, see "Using the RDBMSResolver component (Flash Professional only)" in *Using Components*.

The following example shows an RDBMSResolver component's XML update packet generated with `updateMode` parameter set to `umUsingKey`:

```
<update_packet tableName="customers" nullValue="{_NULL_}"
  transID="46386292065:Wed Jun 25 15:52:34 GMT-0700 2003">
      <delete id="11295627477">
         <field name="id" type="numeric" oldValue="10" key="true"/>
      </delete>
      <insert id="12345678901">
         <field name="id" type="numeric" newValue="20" key="true"/>
         <field name="firstName" type="string" newValue="Davey" key="false"/>
         <field name="lastName" type="string" newValue="Jones" key="false"/>
      </insert>
      <update id="98765432101"> <field name="id" type="numeric" oldValue="30"
  key="true"/>
         <field name="firstName" type="string" oldValue="Peter"
  newValue="Mickey" key="false"/>
       <field name="lastName" type="string" oldValue="Tork" newValue="Dolenz"
  key="false"/>
      </update>
   </update_packet>
```

Elements in the XML update packet include the following:

- `transID`: An ID generated by the DeltaPacket that uniquely identifies this transaction. This information should accompany the results packet returned to this component.
- `delete`: This type of node contains information about a row that was deleted.
- `insert`: This type of node contains information about a row that was added.
- `update`: This type of node contains information about a row that was modified.
- `id`: A number that uniquely identifies the operation within the transaction. This information should accompany the results packet returned to this component.
- `newValue`: This attribute contains the new value for a field that was modified. It appears only when the field value has changed.
- `key`: This attribute is `true` if the field should be used to locate the row to update. This value is determined by the combination of the RDBMSResolver component's `updateMode` parameter, the `fieldInfo.isKey` setting, and the type of operation (insert, delete, update).

The following table describes how the key attributes value is determined. If a field is defined as a key field, using the RDBMSResolver component's `fieldInfo` parameter, it will always appear in the update packet with `key="true"`. Otherwise, the field's key attribute in the update packet will be set according to the following table:

Node type	umUsingKey	umUsingModified	umUsingAll
delete	`false`	`true`	`true`
insert	`false`	`true`	`false`
update	`false`	`true` if the field was modified; `false` otherwise	`true`

About receiving results from an external data source

This section describes requirements that your server code must fulfill. After the server finishes with the update packet, either successfully or unsuccessfully, it should send back a result packet containing errors or additional updates resulting from the update operation. If there are no messages, the results packet should still be sent, but it will have no operation result nodes.

The following example shows a sample RDBMSResolver component results packet (with both update results and change information nodes):

```
<results_packet nullValue="{_NULL_}" transID="46386292065:Wed Jun 25 15:52:34
  GMT-0700 2003">
      <operation op="delete" id="11295627479" msg="The record could not be
  found"/>
      <delete>
         <field name="id" oldValue="1000" key="true" />
      </delete>
      <insert>
         <field name="id" newValue="20"/>
         <field name="firstName" newValue="Davey"/>
         <field name="lastName" newValue="Jones"/>
      </insert>
     <operation op="update" id="02938027477" msg="Couldn't update employee.">
         <field name="id" curValue="105" msg="Invalid field value" />
      </operation>
      <update>
         <field name="id" oldValue="30" newValue="30" key="true" />
         <field name="firstName" oldValue="Peter" newValue="Mickey"/>
         <field name="lastName" oldValue="Tork" newValue="Dolenz"/>
      </update>
   </results_packet>
```

The results packet contains four types of nodes:

Operation nodes contain the result of operations from the update packet. Each operation node should have the following attributes/child nodes:

- The `op` attribute describes the type of operation that was attempted. Must be insert, delete, or update.
- The `id` attribute holds the ID from the operation node that was sent out

- The optional `msg` attribute contains a message string that describes the problem that occurred when attempting the operation
- Zero, one, or more `field` child nodes give field-level specific information. Each `field` node, at a minimum, should have a `name` attribute that contains the field name, and a `msg` attribute that gives the field-level message. It can also optionally contain a `curValue` attribute that holds the most current value for that field in that row on the server.

Update nodes contain information about records that have been modified since the client was last updated. Update nodes should have `field` child nodes that list the fields that are necessary to uniquely identify the record that was deleted and that describe fields that were modified. Each `field` node should have the following attributes:

- The `name` attribute holds the name of the field
- The `oldValue` attribute holds the old value of the field before it was modified. This attribute is required only when the key attribute is included and set to `true`.
- The `newValue` attribute holds the new value that the field should be given. This attribute should not be included if the field was not modified (that is, the field has been included in the list only because it is a key field).
- The `key` attribute holds a Boolean `true` or `false` value that determines whether this field can be used as a key to locate the corresponding record on the client. This attribute should be included and set to `true` for all key fields. It is optional for all others.

Delete nodes contain information about records that have been deleted since the client was updated. Delete nodes should have `field` child nodes that list the fields that are necessary to uniquely identify the record that was deleted. Each `field` node must have a `name` attribute, an `oldValue` attribute, and a `key` attribute whose value is set to `true`.

Insert nodes contain information about records that have been added since the client was updated. Insert nodes should have `field` child nodes that describe the field values that were set when the record was added. Each `field` node must have a `name` attribute and a `newValue` attribute.

Lazy decoding in the WebServiceConnector component

When the WebServiceConnector component receives multiple records of data from a web service, it translates them into an ActionScript array so they are accessible within your application. Translating multiple records of data from XML/SOAP into ActionScript native data can be a time-consuming process; large data sets become large arrays, and can take seconds or tens of seconds.

To improve performance, the WebServiceConnector component supports a feature called lazy decoding, which defers this translation. With lazy decoding, result values that are arrays are not immediately translated from XML to ActionScript. Instead, the result value passed to the user is a special object that acts similarly to an array and translates the XML data only when it is requested. The effect of this feature is to improve the perceived performance of web services by spreading the workload over a longer period of time.

To request the data, use the `myArray[myIndex]` ActionScript expression, as for any array. You must access the array using numeric indices; that is, `myIndex` must be a number. To iterate over the array, use the following statement:

```
for(var i=0; i < myArray.length; i++);
```

The expression `for(var i in myArray)` won't work in this case.

To control lazy decoding, you use ActionScript. For more information, see "SOAPCall.doLazyDecoding" in *Using Components*.

Transfer objects in the DataSet component

It is important to remember that the DataSet component is a collection of transfer objects. This differs from previous implementations of the component, when it was simply an in-memory cache of data (array of record objects). Transfer objects expose business data from an external data source through public properties or accessor methods. When you load data into the DataSet component, the data is translated into a collection of transfer objects. In the simplest scenario, the DataSet component creates and loads the data into anonymous objects. Each anonymous object implements the TransferObject interface, which is all that is required for the DataSet component to manage the objects. The DataSet component tracks changes made to the data and any method calls that are made on the objects. If methods are called on an anonymous object, nothing happens, because the methods don't exist. However, the DataSet component tracks them in the DeltaPacket, which guarantees that they will be sent to the external data source, where they can be called if appropriate.

In an enterprise solution you could create a client-side ActionScript transfer object that mirrors a server-side transfer object. This client object can implement additional methods for manipulating the data or applying client-side constraints. Developers can use the `itemClassName` parameter of the DataSet component to identify the class name of the client-side transfer object that should be created. In this scenario, the DataSet component generates multiple instances of the specified class and initializes it with the loaded data. When `addItem()` is called on the DataSet component, the `itemClassName` is used to create an empty instance of the client-side transfer object.

If you take the enterprise solution one step further, you could implement a client-side transfer object that uses web services or Flash Remoting. In this scenario, the object makes direct calls on the server in addition to possibly storing the calls in the DeltaPacket.

Note: You can create a custom transfer object for use by the DataSet component by creating a class that implements the TransferObject interface. For more information on the TransferObject interface, see "TransferObject interface" in *Using Components*.

CHAPTER 15
Publishing

When you're ready to deliver Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 content to an audience, you can publish it for playback. By default, the Publish command creates a Flash SWF file and an HTML document that inserts your Flash content in a browser window. The Publish command also creates and copies detection files for Flash 4 and later. If you change publish settings, Flash saves the changes with the document. You can create publish profiles to name and save various configurations for the Publish Settings dialog box, in order to quickly publish documents a variety of ways. After you create a publish profile, you can export it for use in other documents, or for use by others working on the same project. For more information, see "Using publish profiles" on page 327.

If you're publishing content that targets Macromedia Flash Player 4 or later, you can implement Flash Player detection, which checks your user's version of Flash Player. If the user doesn't have the specified version, you can direct the user to an alternate web page. For more information, see "Configuring publish settings for Flash Player detection" on page 318

Flash Player 6 and later supports Unicode text encoding. With Unicode support, users can view multilanguage text, regardless of the language used by the operating system running the player. For more information, see Chapter 13, "Creating Multilanguage Text," on page 235.

You can also publish the FLA file in alternative file formats—GIF, JPEG, PNG, and QuickTime—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 Flash Player installed. When you publish a Flash document (FLA file) in alternative file formats, the settings for each file format are stored with the FLA file.

You can also export the FLA file in several formats. Exporting FLA files is similar to publishing FLA files in alternative file formats, except that the settings for each file format are not stored with the FLA file. For more information, see Chapter 16, "Exporting," on page 345.

As an alternative to using the Publish command, if you're proficient in HTML, you can create a custom HTML document with any HTML editor and include the tags required to display a SWF file. For more information, see "About configuring a web server for Flash" on page 343.

Before you publish your SWF file, it's important to test how the SWF file works using the Test Movie and Test Scene commands.

This chapter contains the following sections:

Playing your Flash SWF files

The Macromedia Flash SWF file format is for deploying Flash content.

You can play Flash content in the following ways:

- In Internet browsers such as Netscape Navigator and Internet Explorer that are equipped with Flash Player 7
- With the Flash Xtra in Director and Authorware
- With the Flash ActiveX control in Microsoft Office and other ActiveX hosts
- As part of a QuickTime video
- As a stand-alone video called a projector

The Flash SWF format is an open standard that is supported by other applications. For more information about Flash file formats, see www.macromedia.com/software/flashplayer.

About publishing secure Flash documents

Flash Player 7 contains several features that help you ensure the security of your Flash documents.

Buffer overrun protection

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

About exact domain matching for sharing data between Flash documents

Flash Player 7 enforces a stricter security model than previous versions of Flash Player. There were two primary changes in the security model between Flash Player 6 and 7:

Exact domain matching Flash Player 6 lets SWF files from similar domains (for example, `www.macromedia.com` and `store.macromedia.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 using nonsecure (non-HTTPS) protocols cannot access content loaded using a secure (HTTPS) protocol, even when both are in exactly the same domain.

For more information about ensuring that Flash content performs as expected with the new security model, see "Flash Player security features" in *Using ActionScript in Flash*.

Publishing Flash documents

To publish a Flash document, you select publish file formats and file format settings with the Publish Settings command. Then you publish the Flash document using the Publish command. The publishing configuration that you specify in the Publish Settings dialog box is saved with the document. You can also create and name a publish profile so that the established publish settings are always available.

Depending on the options you specify in the Publish Settings dialog box, the Publish command creates the following files:

- The Flash SWF file
- Alternate images in a variety of formats that appear automatically when Flash Player is not available (GIF, JPEG, PNG, and QuickTime)
- The supporting HTML document(s) required to show SWF content (or an alternate image) in a browser and control browser setting
- Three HTML files (if you keep the default, Detect Flash Version, selected): the detection file, the content file, and the alternate file
- Stand-alone projector files for Windows and Macintosh computers and QuickTime videos from Flash content (EXE, HQX, or MOV files, respectively)

Note: To alter or update a SWF file created with the Publish command, you must edit the original Flash document and then use the Publish command again to preserve all authoring information. Importing a Flash SWF file into Flash removes some of the authoring information.

For information on publish settings, see "Configuring publish settings for Flash Player detection" on page 318. For general information, see "Specifying publish settings that create HTML documents with embedded Flash content" on page 315.

To set general publish settings for a Flash document:

1. Open the Publish Settings dialog box. Do one of the following:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you specify, see "Using publish profiles" on page 327.

2. In the Publish Settings dialog box, select the option for each file format you want to create.

   The Flash SWF format is selected by default. The HTML format is also selected by default because you need an HTML file for a SWF file to appear in a browser. Tabs corresponding to the selected file formats appear above the current panel in the dialog box (except for Windows or Macintosh projector formats, which have no settings). For more information on publish settings for individual file formats, see the following sections.

3. In the File text box for each selected format, either accept the default filename, which corresponds to the name of the document, or enter a new filename with the appropriate extension (such as .gif for a GIF file and .jpg for a JPEG file).

4. Decide where to publish the files. By default, the files are published in the same location as the FLA file. To change where files are published, click the folder beside the filename and browse to a different location in which to publish the file.

5. To create a stand-alone projector file, select Windows Projector or Macintosh Projector.

   Note: The Windows version of Flash adds the .hqx extension to the filename of a Macintosh projector file. You can create a Macintosh projector using the Windows versions of Flash, but you must use a file translator such as BinHex to make the resulting file appear as an application file in the Macintosh Finder.

6. Click the tab for the format options you want to change. Specify publish settings for each format, as described in the following sections.

7. When you have finished setting options, do one of the following:
   - To generate all the specified files, click Publish.
   - To save the settings with the FLA file and close the dialog box without publishing, click OK.

To publish a Flash document without selecting new publish settings:

- Select File > Publish to create the files in the formats and location specified in the Publish Settings dialog box (either the default settings, the settings you selected previously, or the selected publish profile).

Setting publish options for the Flash SWF file format

When publishing a Flash document, you can set image and sound compression options, and an option to protect your SWF file from being imported. You use the controls in the Flash panel of the Publish Settings dialog box to change the settings.

To set publish options for a Flash document:

1. Open the Publish Settings dialog box. Do one of the following:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you specify, see "Using publish profiles" on page 327.

2. Click the Flash tab and select a Player version from the Version pop-up menu.

   Not all Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 features work in published SWF files that target Flash Player versions earlier than Flash Player 7.

   If you want to specific Flash Player detection, on the HTML tab of the Publish Settings dialog box, you must select Flash Player 4 or later. For more information about Flash Player detection, see "Configuring publish settings for Flash Player detection" on page 318.

3. Select a load order to specify how Flash loads a SWF file's layers for showing the first frame of your SWF file: Bottom Up or Top Down.

   This option controls which parts of the SWF file Flash draws first over a slow network or modem connection.

4. In the ActionScript Version pop-up menu, select either ActionScript 1.0 or 2.0 to reflect the version in your document.

   If you select ActionScript 2.0 and you've created classes, you can click the Settings button to set the relative classpath to class files that differs from the path to default directories set in Preferences. For more information, see "Setting the classpath" on page 314.

5. To enable debugging of the published Flash SWF file, select any of the following options:

   Generate Size Report generates a report listing the amount of data in the final Flash content by file.

   Omit Trace Actions causes Flash to ignore Trace actions (`trace`) in the current SWF file. When you select this option, information from Trace actions does not appear in the Output panel.

   For more information, see "Using the Output panel" in *Using ActionScript in Flash*.

   Protect from Import prevents others from importing a SWF file and converting it back into a FLA document. If you select this option, you can decide to use password protection with your Flash SWF file.

   Debugging Permitted activates the Debugger and allows remote debugging of a Flash SWF file. If you select this option, you can decide to use password protection with your SWF file.

   Compress movie compresses the SWF file to reduce file size and download time. This option is selected by default and is most beneficial when a file is text-intensive or includes a lot of ActionScript. A compressed file plays only in Flash Player 6 or later.

Optimize for Flash Player 6 r65 If you selected Flash Player 6 in the Version pop-up menu, you can select this option to target a release of Flash Player 6. The updated version uses ActionScript register allocation to improve performance. Users must have the same release of Flash Player 6 or later.

6. If you selected either Debugging Permitted or Protect from Import in step 5, you can enter a password in the Password text box. If you add a password, others must enter the password before they can debug or import the SWF file. To remove the password, clear the Password text box.

   For more information on the Debugger, see Chapter 4, "Writing and Debugging Scripts" in *Using ActionScript in Flash*.

7. To control bitmap compression, adjust the JPEG Quality slider or enter a value.

   Lower image quality produces smaller files; higher image quality produces larger files. Try different settings to determine the best trade-off between size and quality; 100 provides the highest quality and least compression.

8. To set the sample rate and compression for all streaming sounds or event sounds in the SWF file, click the Set button next to Audio Stream or Audio Event and select options for Compression, Bit Rate, and Quality in the Sound Settings dialog box. Click OK when you finish.

   Note: A streaming sound plays as soon as enough data for the first few frames downloads; it is synchronized to the Timeline. An event sound does not play until it downloads completely, and it continues to play until explicitly stopped.

   For more information on sound, see Chapter 11, "Working with Sound," on page 201.

9. If you want to use the settings selected in step 8 to override settings for individual sounds selected in the Sound section of the Property inspector, select Override Sound Settings. You might want to select this option to create a smaller low-fidelity version of a SWF file.

   Note: If the Select Override Sound Settings option is deselected, Flash scans all stream sounds in the document (including sounds in imported video) and publishes all stream sounds at the highest individual setting. This can increase file size, if one or more stream sounds has a high export setting.

10. (Flash Professional only) To export sounds suitable for devices, including mobile devices, instead of the original library sound, select Export Device Sounds. For more information, see "Using sounds in Flash documents for mobile devices (Flash Professional only)" on page 213. To save the settings with the current file, click OK.

Setting the classpath

To use an ActionScript class that you've defined, Flash must locate the external ActionScript 2.0 files that contain the class definition. The list of folders in which Flash searches for class definitions is called the *classpath*. Classpaths exist at the global/application level and at the document level. For more information about classpaths, see "Understanding the classpath" in *Using ActionScript in Flash*.

To modify the document-level classpath:

1. Select File > Publish Settings to open the Publish Settings dialog box.
2. Click the Flash tab.
3. Verify that ActionScript 2.0 is selected in the ActionScript Version pop-up menu and click Settings.
4. In the ActionScript Settings dialog box, specify the frame on which the class definition should reside in the Export Frame for classes text box.
5. Do any of the following:
   - To add a folder to the classpath, click the Browse to Path button, browse to the folder you want to add, and click OK.

     You can also click the Add New Path (+) button to add a new line to the Classpath list. Double-click the new line, type a relative or absolute path, and click OK.
   - To edit an existing classpath folder, select the path in the Classpath list, click the Browse to Path button, browse to the folder you want to add, and click OK.

     Alternatively, double-click the path in the Classpath list, type the desired path, and click OK.
   - To delete a folder from the classpath, select the path in the Classpath list and click the Remove from Path button.

Specifying publish settings that create HTML documents with embedded Flash content

Playing Flash content in a web browser requires an HTML document that activates the SWF file and specifies browser settings. This document is generated automatically by the Publish command, from HTML parameters in a template document.

The template document can be any text file that contains the appropriate template variables—including a plain HTML file, one that includes code for special interpreters such as ColdFusion or Active Server Pages (ASP), or a template included with Flash (for more information, see "About configuring a web server for Flash" on page 343).

You can customize a built-in template (see "Customizing HTML publishing templates" on page 330) or manually enter HTML parameters for Flash using any HTML editor (see "Editing Flash HTML settings" on page 334).

HTML parameters determine where the Flash content appears in the window, the background color, the size of the SWF file, and so on, and set attributes for the `object` and `embed` tags. You can change these and other settings in the HTML panel of the Publish Settings dialog box. Changing these settings overrides options you've set in the SWF file.

To publish HTML that displays your Flash SWF file:

1. Do one of the following to open the Publish Settings dialog box:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you'll specify, see "Using publish profiles" on page 327.

2. On the Formats tab, the HTML file type is selected by default. In the File text box for the HMTL file, either use the default filename, which matches the name of your document, or enter a unique name, including the .html extension.
3. Click the HTML tab to show HTML settings and select an installed template to use from the Template pop-up menu. Then click the Info button to the right to show a description of the selected template. The default selection is Flash Only.
4. If, in the previous step, you selected an HTML template other than Image Map or QuickTime, and on the Flash tab, you set the Version to Flash Player 4 or later, you can select Flash Version Detection.

   Note: Flash Version Detection configures your document to detect the version of Flash Player that the user has and sends the user to an alternate HTML page if the user does not have the targeted player. For more information on version detection, see "Configuring publish settings for Flash Player detection" on page 318.

5. Select a Dimensions option to set the values of the `width` and `height` attributes in the `object` and `embed` tags:

   Match Movie (the default) uses the size of the SWF file.

   Pixels enters the number of pixels for the width and height in the Width and Height field.

   Percent specifies the percentage of the browser window that the SWF file will occupy.

6. Select Playback options to control the SWF file's playback and features, as described in the following list:

   Paused at Start pauses the SWF file until a user clicks a button or selects Play from the shortcut menu. By default, the option is deselected and the Flash content begins to play as soon as it is loaded (the `PLAY` parameter is set to `true)`.

   Loop repeats the Flash content when it reaches the last frame. Deselect this option to stop the Flash content when it reaches the last frame. (The `LOOP` parameter is on by default.)

   Display Menu shows a shortcut menu when users right-click (Windows) or Control-click (Macintosh) the SWF file. Deselect this option to show only About Flash in the shortcut menu. By default, this option is selected (the `MENU` parameter is set to `true)`.

   Device Font (Windows only) substitutes anti-aliased (smooth-edged) system fonts for fonts not installed on the user's system. Using device fonts increases the legibility of type at small sizes and can decrease the SWF file's size. This option affects only SWF files that contain static text (text that you create when authoring a SWF file and that does not change when the Flash content appears) set to display with device fonts. For more information, see "Using device fonts (static horizontal text only)" on page 115.

7. Select Quality options to determine the trade-off between processing time and appearance, as described in the following list. This option sets the `QUALITY` parameter's value in the `object` and `embed` tags.

   Low favors playback speed over appearance and does not use anti-aliasing.

   Auto Low 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.

   Auto High 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.

   Medium applies some anti-aliasing but does not smooth bitmaps. It produces a better quality than the Low setting but lower quality than the High setting.

   High (the default) favors appearance over playback speed and always uses anti-aliasing. If the SWF file does not contain animation, bitmaps are smoothed; if the SWF file has animation, bitmaps are not smoothed.

   Best provides the best display quality and does not consider playback speed. All output is anti-aliased and bitmaps are always smoothed.

8. Select a Window Mode option, which controls the HTML `wmode` attribute in the `object` and `embed` tags. The window mode modifies the relationship of the Flash content bounding box or virtual window with content in the `HTML` page as described in the following list:

   Window does not embed any window-related attributes in the object and embed tags. The background of the Flash content is opaque and uses the HTML background color. The HTML cannot render above or below the Flash content. This is the default setting.

   Opaque Windowless sets the background of the Flash content to opaque, obscuring anything underneath the Flash content. Opaque Windowless lets HTML content appear above or on top of Flash content.

   Transparent Windowless sets the background of the Flash content to transparent. This allows the HTML content to appear above and below the Flash content.

   Note: In some instances, complex rendering in Transparent Windowless mode can result in slower animation when the HTML images are also complex.

   See the table following this procedure for browsers that support windowless modes.

9. Select one of the following HTML Alignment options to position the Flash SWF window in the browser window:

   Default centers the Flash content in the browser window and crops edges if the browser window is smaller than the application.

   Left, **Right**, **Top**, or **Bottom** align SWF files along the corresponding edge of the browser window and crop the remaining three sides as needed.

10. Select a Scale option to place the Flash content within specified boundaries, if you've changed the document's original width and height. The Scale option sets the `SCALE` parameter in the `object` and `embed` tags.

    Default (Show All) shows the entire document in the specified area without distortion while maintaining the original aspect ratio of the SWF files. Borders can appear on two sides of the application.

    No Border scales the document to fill the specified area and keeps the SWF file's original aspect ratio without distortion, cropping the SWF file if needed.

    Exact Fit displays the entire document in the specified area without preserving the original aspect ratio, which can cause distortion.

    No Scale prevents the document from scaling when the Flash Player window is resized.

11. Select a Flash Alignment option to set how the Flash content is placed within the application window and how it is cropped, if necessary. This option sets the `SALIGN` parameter of the `object` and `embed` tags.
    - For Horizontal alignment, select Left, Center, or Right.
    - For Vertical alignment, select Top, Center, or Bottom.
12. Select Show Warning Messages to show error messages if tag settings conflict—for example, if a template has code referring to an alternate image that has not been specified.
13. To save the settings with the current file, click OK.

The following browsers support windowless modes:

Operating system	Internet Explorer	Netscape	Other
Macintosh OS X 10.1.5 and 10.2	5.1 and IE 5.2	7.0 and later	• Opera 6 or later • Mozilla 1.0 or later • AOL/Compuserve
Windows	5.0, 5.5, and 6.0	7.0 and later	• Opera 6 and later • Mozilla 1.0 and later • AOL/Compuserve

Configuring publish settings for Flash Player detection

You can configure your document to detect your users' Flash Player version. If you've selected Detect Flash Version in the Publish Settings dialog box, users who access your Flash application are directed transparently to an HTML file that contains a SWF file designed to detect their Flash Player version. If they have the specified version or later, the SWF file again redirects the user to your content HTML file, and your SWF file plays as designed. If users don't have the specified version, they're redirected to an alternate HTML file that Flash creates or that you've created.

To enable Flash Player detection:

1. Select Detect Flash Version on the HTML tab of the Publish Settings dialog box. For general information, see "Specifying publish settings that create HTML documents with embedded Flash content" on page 315.

   Note: The option is available only if you selected Flash Player 4 or later on the Flash tab of the Publish Settings dialog box, and if you have not selected QuickTime or Image Map as a template.

2. Click Settings for Detect Flash Version. The dialog box shows the Flash Player version that you selected on the Flash tab of the Publish Settings dialog box. You can use the Major Revision and Minor Revision text boxes to specify precise revisions of Flash Player.

3. The Detection File text box shows the name of the HTML file that contains the SWF file designed to detect the Player version and redirect users to the appropriate HTML page. You can accept the default name or change it.

   Note: Changing the default name also changes the name in the HTML text box on the Formats tab of the Publish Settings dialog box.

4. Use the Content File text box to specify the name of the HTML template that contains your Flash content. The default name is the name of your document, appended with _content.

5. Do one of the following to create the alternate HTML page, for users who don't have the specified Flash Player version:

   - If you want Flash to automatically create an alternate HTML file, select Generate Default and either accept the default filename in the Alternate File text box or enter a new filename.
   - If you created an HTML file to use as the alternate file, select Use Existing, click Browse, and select the HTML file.

6. Click OK to return to the Publish Settings dialog box.

Specifying publish settings for GIF files

GIF files provide an easy way to export drawings and simple animations for use in web pages. Standard GIF files are simply compressed bitmaps.

An animated GIF file (sometimes referred to as a GIF89a) offers a simple way to export short animation sequences. Flash optimizes an animated GIF file, storing only frame-to-frame changes.

Flash exports the first frame in the SWF file as a GIF file, unless you mark a different keyframe for export by entering the frame label #Static in the Property inspector. Flash exports all the frames in the current SWF file to an animated GIF file unless you specify a range of frames for export by entering the frame labels #First and #Last in the appropriate keyframes.

Flash can generate an image map for a GIF file to maintain URL links for buttons in the original document. You use the Property inspector to place the frame label #Map in the keyframe in which you want to create the image map. If you don't create a frame label, Flash creates an image map using the buttons in the last frame of the SWF file. You can create an image map only if the `$IM` template variable is present in the template you select. For more information, see "Creating an image map" on page 332.

To publish a GIF file with your Flash file:

1. Do one of the following to open the Publish Settings dialog box:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you specify, see "Using publish profiles" on page 327.
2. On the Formats tab, select the GIF Image type. In the File text box for the GIF image, either use the default filename or enter a new filename with the .gif extension.
3. Click the GIF tab to show the file settings.
4. For Dimensions, enter a width and height in pixels for the exported bitmap image, or select Match Movie to make the GIF the same size as the Flash SWF file and maintain the aspect ratio of your original image.
5. Select a Playback option to determine whether Flash creates a still (Static) image or an animated GIF (Animation). If you select Animation, select Loop Continuously or enter the number of repetitions.
6. Select one of the following options to specify a range of appearance settings for the exported GIF file:

   Optimize Colors removes any unused colors from a GIF file's color table. This option reduces the file size by 1000 to 1500 bytes without affecting image quality, but slightly increases the memory requirements. This option has no effect on an adaptive palette. (An adaptive palette analyzes the colors in the image and creates a unique color table for the selected GIF file.)

   Interlace incrementally shows the exported GIF file in a browser as it downloads. Interlacing lets the user see basic graphic content before the file has completely downloaded and can download the file faster over a slow network connection. Do not interlace an animated GIF image.

   Smooth applies anti-aliasing to an exported bitmap to produce a higher-quality bitmap image and improve text display quality. However, smoothing might cause a halo of gray pixels to appear around an anti-aliased image placed on a colored background, and it increases the GIF file size. Export an image without smoothing if a halo appears or if you're placing a GIF transparency on a multicolored background.

   Dither Solids applies dithering to solid colors as well as gradients. For more information, see Dither options in step 8.

   Remove Gradients, which is turned off by default, converts all gradient fills in the SWF file to solid colors using the first color in the gradient. Gradients increase the size of a GIF file and are often poor quality. If you use this option, select the first color of your gradients carefully to prevent unexpected results.

7. Select one of the following Transparent options to determine the transparency of the application's background and the way alpha settings are converted to GIF:

   Opaque makes the background a solid color.

   Transparent makes the background transparent.

   Alpha sets partial transparency. You can enter a Threshold value between 0 and 255. A lower value results in greater transparency. A value of 128 corresponds to 50% transparency.

8. Select a Dither option to specify how pixels of available colors are combined to simulate colors not available in the current palette. Dithering can improve color quality, but it increases the file size. Select from the following options:

   None turns off dithering and replaces colors not in the basic color table with the solid color from the table that most closely approximates the specified color. Turning dithering off can result in smaller files but unsatisfactory colors.

   Ordered provides good-quality dithering with the smallest increase in file size.

   Diffusion provides the best-quality dithering but increases file size and processing time. It also works only with the web 216 color palette selected.

9. Select one of the following Palette Types to define the image's color palette:

   Web 216 uses the standard 216-color, browser-safe palette to create the GIF image, for good image quality and the fastest processing on the server.

   Adaptive analyzes the colors in the image and creates a unique color table for the selected GIF file. This option is best for systems displaying thousands or millions of colors; it creates the most accurate color for the image but increases file size. To reduce the size of a GIF with an adaptive palette, use the Max Colors option in step 10 to decrease the number of colors in the palette.

   Web Snap Adaptive is the same as the Adaptive palette option except it converts similar colors to the web 216 color palette. The resulting color palette is optimized for the image, but when possible, Flash uses colors from the web 216 palette. This produces better colors for the image when the web 216 palette is active on a 256-color system.

   Custom specifies a palette that you have optimized for the selected image. The custom palette is processed at the same speed as the web 216 palette. To use this option, you should know how to create and use custom palettes. To select a custom palette, click the Ellipsis (...) button to the right of the Palette box at the bottom of the dialog box and select a palette file. Flash supports palettes saved in the ACT format, exported by Macromedia Fireworks and other leading graphics applications; for more information, see "Importing and exporting color palettes" on page 83.

10. If you selected the Adaptive or Web Snap Adaptive palette in step 9, enter a value for Max Colors to set the number of colors used in the GIF image. Selecting a smaller number of colors can produce a smaller file but can degrade the colors in the image.

11. Click OK to save the settings with the current file.

Specifying publish settings for JPEG files

The JPEG format lets you save an image as a highly compressed, 24-bit bitmap. Generally, GIF format is better for exporting line art, and JPEG format is better for images with continuous tones, such as photographs, gradients, or embedded bitmaps.

Flash exports the first frame in the SWF file as a JPEG, unless you mark a different keyframe for export by entering the frame label **#Static**.

To publish a JPEG file with your Flash SWF file:

1. Do one of the following to open the Publish Settings dialog box:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you specify, see "Using publish profiles" on page 327.
2. On the Formats tab, select the JPEG Image type. For the JPEG filename, either use the default filename, or enter a new filename with the .jpg extension.
3. Click the JPEG panel to show its settings.
4. For Dimensions, enter a width and height in pixels for the exported bitmap image, or select Match Movie to make the JPEG image the same size as the Stage and maintain the aspect ratio of your original image.
5. For Quality, drag the slider or enter a value to control the amount of JPEG file compression.

   The lower the image quality, the smaller the file size, and vice versa. Try different settings to determine the best compromise between size and quality.

   Note: You can set the bitmap export quality per object using the Bitmap Properties dialog box to change the object's compression setting. Selecting the default compression option in the Bitmap Properties dialog box applies the Publish Settings JPEG Quality option. For more information, see "Setting bitmap properties" on page 139.
6. Select Progressive to show Progressive JPEG images incrementally in a web browser, which makes images appear faster when loading with a slow network connection.

   This option is similar to interlacing in GIF and PNG images.
7. To save the settings with the current file, click OK.

Specifying publish settings for PNG files

PNG is the only cross-platform bitmap format that supports transparency (an alpha channel). It is also the native file format for Macromedia Fireworks.

Flash exports the first frame in the SWF file as a PNG file, unless you mark a different keyframe for export by entering the frame label **#Static**.

To publish a PNG file with your Flash SWF file:

1. Do one of the following to open the Publish Settings dialog box:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you specify, see "Using publish profiles" on page 327.

2. On the Formats tab, select the PNG Image type. For the PNG filename, either use the default filename, or enter a new filename with the .png extension.
3. Click the PNG tab. For Dimensions, enter values for width and height in pixels for the exported bitmap image, or select Match Movie to make the PNG image the same size as the Flash SWF file and maintain the aspect ratio of your original image.
4. Select a bit depth to set the number of bits per pixel and colors to use in creating the image:
   - Select 8-bit for a 256-color image.
   - Select 24-bit for thousands of colors.
   - Select 24-bit with Alpha for thousands of colors with transparency (32 bits).

   The higher the bit depth, the larger the file.
5. Select one of the following options to specify appearance settings for the exported PNG:

   Optimize Colors removes any unused colors from a PNG file's color table. This option reduces the file size by 1000 to 1500 bytes without affecting image quality but increases the memory requirements slightly. This option has no effect on an adaptive palette.

   Interlace incrementally shows the exported PNG in a browser as it downloads. Interlacing lets the user see basic graphic content before the file has completely downloaded and might download the file faster with a slow network connection. Do not interlace an animated PNG file.

   Smooth applies anti-aliasing to an exported bitmap to produce a higher-quality bitmap image and improve text display quality. However, smoothing might cause a halo of gray pixels to appear around an anti-aliased image placed on a colored background, and it increases the PNG file size. Export an image without smoothing if a halo appears or if you're placing a PNG transparency on a multicolored background.

   Dither Solids applies dithering to solid colors and gradients. For more information, see Dither options in step 6.

   Remove Gradients, which is turned off by default, converts all gradient fills in the application to solid colors using the first color in the gradient. Gradients increase the size of a PNG and are often poor quality. If you use this option, select the first color of your gradients carefully to prevent unexpected results.

6. If you selected 8-bit for Bit Depth in step 4, select a Dither option to specify how pixels of available colors are mixed to simulate colors not available in the current palette. Dithering can improve color quality, but it increases file size. Select from the following options:

   None turns off dithering and replaces colors not in the basic color table with the solid color from the table that most closely approximates the specified color. Turning dithering off can produce smaller files but unsatisfactory colors.

   Ordered provides good-quality dithering with the smallest increase in file size.

   Diffusion provides the best-quality dithering but increases file size and processing time. It also works only with the web 216 color palette selected.

7. Select one of the following Palette Types to define the color palette for the PNG image:

   Web 216 uses the standard 216-color, browser-safe palette to create the PNG image, for good image quality and the fastest processing on the server.

   Adaptive analyzes the colors in the image and creates a unique color table for the selected PNG file. This option is best for systems showing thousands or millions of colors; it creates the most accurate color for the image but results in a file size larger than a PNG created with the web 216 palette.

   Web Snap Adaptive is the same as the Adaptive palette option except that it converts very similar colors to the web 216 color palette. The resulting color palette is optimized for the image, but when possible, Flash uses colors from web 216. This produces better colors for the image when the web 216 palette is active on a 256-color system.

   To reduce the size of a PNG file with an adaptive palette, use the Max Colors option to decrease the number of palette colors, as described in the next step.

   Custom specifies a palette that you have optimized for the selected image. The custom palette is processed at the same speed as the web 216 palette. To use this option, you should know how to create and use custom palettes. To select a custom palette, click the Ellipsis (...) button to the right of the Palette box at the bottom of the dialog box and select a palette file. Flash supports palettes saved in the ACT format, exported by Macromedia Fireworks and other leading graphics applications; for more information, see "Importing and exporting color palettes" on page 83.

8. If you selected the Adaptive or Web Snap Adaptive palette in step 7, enter a value for Max Colors to set the number of colors used in the PNG image. Selecting a smaller number of colors can produce a smaller file but might degrade the colors in the image.

9. Select one of the following Filter Options to select a line-by-line filtering method to make the PNG file more compressible, and experiment with the different options for a particular image:

   None turns off filtering.

   Sub transmits the difference between each byte and the value of the corresponding byte of the prior pixel.

   Up transmits the difference between each byte and the value of the corresponding byte of the pixel immediately above.

   Average uses the average of the two neighboring pixels (left and above) to predict the value of a pixel.

Path computes a simple linear function of the three neighboring pixels (left, above, upper left), and then selects the neighboring pixel closest to the computed value as a predictor.

Adaptive analyzes the colors in the image and creates a unique color table for the selected PNG file. This option is best for systems showing thousands or millions of colors; it creates the most accurate color for the image but results in a file size larger than a PNG created with the web 216 palette. You can reduce the size of a PNG created with an adaptive palette by decreasing the number of colors in the palette.

10. To save the settings with the current file, click OK.

Specifying publish settings for QuickTime videos

The QuickTime Publish Settings option creates videos in the same QuickTime format you have installed on your computer. For example, if you have QuickTime 5 installed, Flash publishes the QuickTime video in version 5.

The Flash document plays in the QuickTime video exactly as it does in Flash Player, retaining all its interactive features. If the Flash document also contains a QuickTime video, Flash copies it to its own track in the new QuickTime file.

The current version of the QuickTime Player (as of this writing) supports Flash Player 4 SWF file playback. For best results, Flash content that you export to the QuickTime format should contain only those features supported by Flash Player 4. Future releases of the QuickTime Player might support additional Flash file formats.

If you try to export Flash Player 6 or 7 content to the QuickTime format, an error message will appear, indicating that the installed version of QuickTime does not support that version of Flash Player. To resolve this issue, you can select Flash Player 4 from the Version pop-up menu on the Flash tab of the Publish Settings dialog box. For more information, see "Setting publish options for the Flash SWF file format" on page 312.

If a newer version of the QuickTime Player becomes available that supports Flash Player 6 and later versions, you can install the updated QuickTime version and publish your document as QuickTime files that target those versions of Flash Player.

For more information on QuickTime videos, see your QuickTime documentation.

To publish a QuickTime video with your Flash SWF file:

1. Do one of the following to open the Publish Settings dialog box:
   - Select File > Publish Settings.
   - In the Property inspector for the document (which is available when no object is selected), click the Settings button.

   Note: To create a publish profile for the publish settings that you specify, see "Using publish profiles" on page 327.

2. On the Formats tab, select the QuickTime file type. For the QuickTime filename, either use the default filename, or enter a new filename with the .mov extension.
3. Click the QuickTime panel to show its settings.

4. For Dimensions, enter a width and height in pixels for the exported QuickTime video, or select Match Movie to make the QuickTime video the same size as the Flash SWF file and keep its aspect ratio.
5. Select one of the following Alpha options to control the transparency (alpha) mode of the Flash track in the QuickTime video without affecting any alpha settings in the Flash application:

   Alpha Transparent makes the Flash track SWF file transparent and shows any content in tracks behind the Flash track.

   Copy makes the Flash track opaque and masks all content in tracks behind the Flash track.

   Auto makes the Flash track transparent if it is on top of any other tracks, but opaque if it is the bottom or only track in the SWF file.
6. Select one of the following Layer options to control where the Flash track plays in the stacking order of the QuickTime video:

   Top places the Flash track always on top of other tracks in the QuickTime video.

   Bottom places the Flash track always behind other tracks.

   Auto places the Flash track in front of other tracks if Flash objects are in front of video objects in the Flash application, and behind all other tracks if Flash objects are not in front.
7. Select Streaming Sound to have Flash export all the streaming audio in the Flash SWF file to a QuickTime sound track, recompressing the audio using the standard QuickTime audio settings. To change these options, click Audio Settings; for more information, see your QuickTime documentation.
8. Select Controller to specify the type of QuickTime controller used to play the exported video—None, Standard, or QuickTime VR.
9. Select one of the following Playback options to control how QuickTime plays a video:

   Looping repeats the video when it reaches the last frame.

   Paused at Start pauses the video until a user clicks a button in the video or selects Play from the shortcut menu. By default, the option is deselected; that is, the video begins to play as soon as it is loaded.

   Play Every Frame shows every frame of the video without skipping to maintain time and does not play sound.
10. Select File Flatten (Make Self-Contained) to combine the Flash content and imported video content into a single QuickTime video. Deselecting this option makes the QuickTime video refer to the imported files externally; the video won't work properly if these files are missing.
11. To save the settings with the current file, click OK.

About publishing Flash Lite documents

Macromedia Flash Lite lets Flash designers, developers, and content providers quickly create engaging content for mobile phones using the ActionScript scripting language, drawing tools, and templates. For detailed information on authoring for mobile devices, see the Content Development Kits on the Mobile and Devices Development Center at www.macromedia.com/devnet/devices.

Flash Lite 1.1 supports two new ActionScript functions: `FSCommand()` and `FSCommand2()`. Many new FSCommand and FSCommand2 commands have been introduced in Flash Lite 1.1. For a complete list of ActionScript expressions supported, see the FlashLite1.1_Authoring_Guidelines.pdf file on your Macromedia Flash application CD or on the Mobile and Devices Development Center.

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

Using publish profiles

You can create a publish profile that saves a publish settings configuration. You can then export the publish profile to other documents or for others to use them. Conversely, you can import publish profiles to use in your document. Publish profiles offer many advantages, including the following:

- You can create profiles to publish in several media formats.
- You can create a publish profile for in-house use that differs from the way you'd publish the files for a client.
- Your company can create a standard publish profile to ensure files are published uniformly.

Publish profiles, as with default publish settings, are saved at the document rather than application level. To use a publish profile in another document, you export it and import it into the other file. For more information, see "Exporting a publish profile" on page 328 and "Importing a publish profile" on page 328.

Creating a publish profile

The Publish Settings dialog box includes a Create New Profile button, which creates a profile based on the publish settings you've specified.

To create a publish profile:

1. In the Publish Settings dialog box, click the Create New Profile (+) button.
2. In the Create New Profile dialog box, name the publish profile, and click OK.

   The newly created publish profile appears as a selection in the Current Profile pop-up menu of the Publish Settings dialog box.
3. Specify the publish settings for your document in the Publish Settings dialog box (File > Publish Settings), and click OK. For more information about configuring publish settings, see "Publishing Flash documents" on page 311.

Duplicating a publish profile

If you've modified publish settings for a publish profile and you want to save the modifications, you can create a duplicate profile.

To duplicate a publish profile:

1. From the Current Profile pop-up menu in the Publish Settings dialog box (File > Publish Settings), select the publish profile that you want to copy.
2.  Click the Duplicate Profile button.
3. In the Duplicate Profile dialog box, enter the profile name in the Duplicate Name text box, and click OK.

   The duplicate publish profile appears as a selection in the Current Profile pop-up menu of the Publish Settings dialog box.

Modifying a publish profile

To modify a publish profile, you simply change the settings in the Publish Settings dialog box.

To modify a publish profile:

1. From the Current Profile pop-up menu in the Publish Settings dialog box (File > Publish Settings), select the publish profile that you want to copy.
2. Specify the new publish settings for your document, and click OK. For details about how to select options in the dialog box, see "Publishing Flash documents" on page 311.

Exporting a publish profile

You can export a publish profile as an XML file for import into other documents. After import, the publish profile appears in the Publish Settings dialog box as an option in the Current Profile pop-up menu.

To export a publish profile:

1. From the Current Profile pop-up menu in the Publish Settings dialog box (File > Publish Settings), select the publish profile that you want to export.
2. Click the Import/Export Profile button, and select Export.
3. In the Export Profile dialog box, either accept the default location in which to save the publish profile or browse to a new location and click Save.

Importing a publish profile

Other users can create and export publish profiles, which you can import and select as a publish settings option.

To import a publish profile:

1. In the Publish Settings dialog box (File > Publish Settings), click Import/Export Profile, and select Import.
2. In the Import Profile dialog box, browse to the publish profile XML file, and click Open.

Deleting a publish profile

When you no longer need a publish profile, you can delete it from the document.

To delete a publish profile:

1. In the Publish Settings dialog box (File > Publish Settings), select the publish profile that you want to delete in the Current Profile pop-up menu.
2.  Click the Delete Profile button. In the dialog box that requests confirmation of the deletion, click OK.

About HTML publishing templates

A Flash HTML template is a text file that contains both unchanging HTML code and template code or variables (which differ from ActionScript variables). When you publish a Flash SWF file, Flash replaces the variables in the template you selected in the Publish Settings dialog box with your HTML settings and produces an HTML page with your SWF file embedded.

Flash includes various templates, suitable for most users' needs, that eliminate the need to edit an HTML page with the Flash SWF file. For example, one template simply places a Flash SWF file on the generated HTML page so that users can view it through a web browser if the plug-in is installed.

You can easily use the same template, change the settings, and publish a new HTML page. If you're proficient in HTML, you can also create custom templates using any HTML editor. Creating a template is the same as creating a standard HTML page, except that you replace specific values pertaining to a Flash SWF file with variables that begin with a dollar sign ($).

Flash HTML templates have the following characteristics:

- A one-line title that appears on the Template pop-up menu.
- A longer description that appears when you click the Info button.
- Template variables beginning with a dollar sign ($) specify where parameter values should be substituted when Flash generates the output file.

  Note: Use a backslash and dollar sign (\ $) combination if you need to use a dollar sign for another purpose in the document.

- HTML `object` and `embed` tags that follow the tag requirements of Microsoft Internet Explorer and Netscape Communicator/Navigator, respectively. To display a SWF file properly on an HTML page, you must follow these tag requirements. Internet Explorer opens a Flash SWF file using the `object` HTML tag; Netscape uses the `embed` tag. For more information, see "Using object and embed tags" on page 334.

Customizing HTML publishing templates

If you're familiar with HTML, you can modify HTML template variables to create an image map, a text report or a URL report, or to insert custom values for some of the most common Flash `object` and `embed` parameters (for Internet Explorer and Netscape Communicator/Navigator, respectively).

Flash templates can include any HTML content for your application or even code for special interpreters such as ColdFusion and ASP.

To modify an HTML publishing template:

1. Using an HTML editor, open the Flash HTML template you want to change. These templates can be found in the following locations:

   For Windows operating systems:

   Windows 2000 or XP <boot drive>:\Documents and Settings\<user>\Local Settings\Application Data\Macromedia\Flash MX 2004\<language>\Configuration\HTML

   - The <boot drive> is the drive from which 2000 or XP boots (usually C).
   - The <user> is the name of the person logged in to the 2000 or XP.
   - The <language> is set to an abbreviated language name. For example, in the US, <language> is set to "en" for English.

   Note: The Application Data folder is normally a hidden folder; you might need to change your Windows Explorer settings to see this folder.

   Windows 98 *boot drive*:\Program Files\Macromedia\Flash MX 2004*language*\First Run\HTML

   For Macintosh operating systems:

   Macintosh OS X 10.1.5 and 10.2.6 and later *Macintosh HD*/Applications/Macromedia Flash MX 2004/First Run/HTML

2. Edit the template as needed.
   - For information on variables that Flash supports, see the table that follows this procedure.
   - For information on creating an image map or a text or URL report, or to insert custom values for `object` and `embed` parameters, see the sections for those topics, which follow this procedure.
3. When you finish editing the variables, save the template in the same folder from which you retrieved it.
4. To apply the template settings to your Flash SWF file, select File > Publish Settings, select the HTML panel, and select the template you modified.

   Flash changes only the template variables in the template selected in the Publish Settings dialog box.
5. Select your remaining publish settings, and click OK. For more information, see "Publishing Flash documents" on page 311.

Using HTML template variables

The following table lists the template variables that Flash recognizes. For a definition of all the tags with which these variables work, see "Editing Flash HTML settings" on page 334.

Attribute/parameter	Template variable
Template title	$TT
Template description start	$DS
Template description finish	$DF
Flash (SWF file) title	$T1
Width	$WI
Height	$HE
Movie	$MO
HTML alignment	$HA
Looping	$LO
Parameters for object	$PO
Parameters for embed	$PE
Play	$PL
Quality	$QU
Scale	$SC
Salign	$SA
Wmode	$WM
Devicefont	$DE
Bgcolor	$BG
Movie text (area to write movie text)	$MT
Movie URL (location of SWF file URL)	$MU
Image width (unspecified image type)	$IW
Image height (unspecified image type)	$IH
Image filename (unspecified image type)	$IS
Image map name	$IU
Image map tag location	$IM
QuickTime width	$QW
QuickTime height	$QH
QuickTime filename	$QN
GIF width	$GW
GIF height	$GH
GIF filename	$GN
JPEG width	$JW

Attribute/parameter	Template variable
JPEG height	$JH
JPEG filename	$JN
PNG width	$PW
PNG height	$PH
PNG filename	$PN

Creating an image map

Flash can generate an image map to show any image and maintain the function of buttons that link to URLs. When an HTML template includes the $IM template variable, Flash inserts the image map code. The $IU variable identifies the name of the GIF, JPEG, or PNG file.

To create an image map:

1. In your Flash document, select the keyframe to be used for the image map and label it **#Map** in the frame Property inspector (select Window > Properties if the Property inspector is not visible). You can use any keyframe with buttons that have attached Get URL actions.

   If you don't create a frame label, Flash creates an image map using the buttons in the last frame of the SWF file. This option generates an embedded image map, not an embedded Flash SWF file.
2. To select the frame to show the image map, do one of the following:
   - For PNG or GIF files, label the frame to appear as **#Static**.
   - For JPEG, during the publish operation, place the playhead on the frame to be used for display.
3. In an HTML editor, open the HTML template you'll modify. Flash stores HTML templates in the following location: <boot drive>:\Program Files\Macromedia\Flash MX 2004\<language>\First Run\HTML.
4. Save your template.
5. Select File > Publish Settings, click the Format tab, and select a format for the image map: GIF, JPEG, or PNG.
6. Click OK to save your settings.

   For example, insert the following code in a template:

```
$IM
<img src=$IS usemap=$IU width=$IW height=$IH BORDER=0>
```

   This might produce the following code in the HTML document created by the Publish command:

```
<map name="mymovie">
<area coords="130,116,214,182" href="http://www.macromedia.com">
</map>
<img src="mymovie.gif" usemap="#mymovie" width=550 height=400 border=0>
```

Creating a text report

The $MT template variable causes Flash to insert all the text from the current Flash SWF file as a comment in the HTML code. This is useful for indexing the content of a SWF file and making it visible to search engines.

Creating a URL report

The $MU template variable makes Flash generate a list of the URLs referred to by actions in the current SWF file and insert it at the current location as a comment. This lets link verification tools see and verify the links in the SWF file.

Using shorthand template variables

The $PO (for object tags) and $PE (for embed tags) template variables are useful shorthand elements. Each variable causes Flash to insert into a template any nondefault values for some of the most common Flash object and embed parameters, including PLAY ($PL), QUALITY ($QU), SCALE ($SC), SALIGN ($SA), WMODE ($WM), DEVICEFONT ($DE), and BGCOLOR ($BG). For an example of these variables, see the sample template in the following section.

Sample template

The following Default.HTML template file in Flash includes many of the commonly used template variables:

```
$TTFlash Only
$DS
Display Macromedia Flash Movie in HTML.
$DF

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
  www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
$CS
<title>$TI</title>
</head>
<body bgcolor="$BG">
<!--url's used in the movie-->
$MU
<!--text used in the movie-->
$MT
<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000" codebase="http://
  fpdownload.macromedia.com/pub/shockwave/cabs/flash/
  swflash.cab#version=7,0,0,0" width="$WI" height="$HE" id="$TI" align="$HA">
<param name="allowScriptAccess" value="sameDomain" />
$PO
<embed $PEwidth="$WI" height="$HE" name="$TI" align="$HA"
  allowScriptAccess="sameDomain" type="application/x-shockwave-flash"
  pluginspage="http://www.macromedia.com/go/getflashplayer" />
</object>
</body>
</html>
```

Editing Flash HTML settings

You need an HTML document to play a Flash SWF file in a web browser and specify browser settings. If you are experienced with HTML, you can change or enter HTML parameters in an HTML editor or create custom HTML files to control a Flash SWF file.

You can also have Flash create the HTML document automatically when you publish a SWF file; see "Publishing Flash documents" on page 311. For information on customizing HTML templates included in Flash, see "Customizing HTML publishing templates" on page 330.

Using object and embed tags

To display a Flash SWF file in a web browser, an HTML document must use the `object` and `embed` tags with the proper parameters.

For `object`, four settings (`height`, `width`, `classid`, and `codebase`) are attributes that appear within the `object` tag; all others are parameters that appear in separate, named `param` tags, as shown in the following example:

```
<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000" width="100"
height="100" codebase="http://fpdownload.macromedia.com/pub/shockwave/cabs/
  flash/swflash.cab#version=7,0,0,0">
<param name="movie" value="moviename.swf">
<param name="play" value="true">
<param name="loop" value="true">
<param name="quality" value="high">
</object>
```

For the `embed` tag, all settings (such as `height`, `width`, `quality`, and `loop`) are attributes that appear between the angle brackets of the opening `embed` tag, as shown in the following example:

```
<embed src="moviename.swf" width="100" height="100" play="true"
loop="true" quality="high"
pluginspage="http://www.macromedia.com/shockwave/download/
  index.cgi?P1_Prod_Version=ShockwaveFlash">
</embed>
```

To use both tags, position the `embed` tag before the closing `object` tag, as shown in the following example:

```
<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000" width="100"
height="100" codebase="http://fpdownload.macromedia.com/pub/shockwave/cabs/
  flash/swflash.cab#version=7,0,0,0">
<param name="movie" value="moviename.swf">
<param name="play" value="true">
<param name="loop" value="true">
<param name="quality" value="high">

<embed src="moviename.swf" width="100" height="100" play="true"
loop="true" quality="high"
pluginspage="http://www.macromedia.com/shockwave/download/
  index.cgi?P1_Prod_Version=ShockwaveFlash">
</embed>

</object>
```

Note: If you use both the `object` and the `embed` tags, use identical values for each attribute or parameter to ensure consistent playback across browsers. The parameter `swflash.cab#version=7,0,0,0` is optional, and you can omit it if you don't want to check for the version number.

Parameters and attributes

The following tag attributes and parameters describe the HTML code created by the Publish command. You can refer to this list as you write custom HTML to show Flash content. Unless noted, all items apply to both the `object` and `embed` tags. Optional entries are noted. Parameters are used with the `object` tag and are recognized by Internet Explorer, but the `embed` tag is recognized by Netscape. Attributes are used with both the `object` and `embed` tags. When customizing a template, you can substitute a template variable (identified in the Value section for each parameter in the list that follows) for the value. For more information, see "Customizing HTML publishing templates" on page 330.

Note: The attributes and parameters listed in this section are shown in lowercase letters purposely to comply with the XHTML standard.

devicefont attribute/parameter

Value

`true | false`

Template variable: `$DE`

Description

(Optional) Specifies whether static text objects for which the Device Font option is not selected will be drawn using a device font anyway, if the fonts needed are available from the operating system.

src attribute

Value

`movieName``.swf`

Template variable: `$MO`

Description

Specifies the name of the SWF file to be loaded. Applies to the `embed` tag only.

movie parameter

Value

`movieName``.swf`

Template variable: `$MO`

Description

Specifies the name of the SWF file to be loaded. Applies to the `object` tag only.

classid attribute

Value

`clsid:d27cdb6e-ae6d-11cf-96b8-444553540000`

Description

Identifies the ActiveX control for the browser. The value must be entered exactly as shown. Applies to the `object` tag only.

width attribute

Value

n or *n%*

Template variable: `$WI`

Description

Specifies the width of the application either in pixels or as a percentage of the browser window.

height attribute

Value

n or *n%*

Template variable: `$HE`

Description

Specifies the height of the application either in pixels or as a percentage of the browser window.

Note: Because Flash applications are scalable, quality doesn't degrade at different sizes if the aspect ratio is maintained. (For example, the following sizes all have a 4:3 aspect ratio: 640 x 480 pixels, 320 x 240 pixels, and 240 x 180 pixels.)

codebase attribute

Value

```
http://fpdownload.macromedia.com/pub/shockwave/cabs/flash/
  swflash.cab#version=7,0,0,0
```

Description

Identifies the location of the Flash Player ActiveX control so that the browser can automatically download it if it is not already installed. The value must be entered exactly as shown. Applies to the `object` tag only.

pluginspage attribute

Value

```
http://www.macromedia.com/shockwave/download/
index.cgi?P1_Prod_Version=ShockwaveFlash
```

Description

Identifies the location of the Flash Player plug-in so that the user can download it if it is not already installed. The value must be entered exactly as shown. Applies to the `embed` tag only.

swliveconnect attribute

Value

`true` | `false`

Description

(Optional) Specifies whether the browser should start Java when loading Flash Player for the first time. The default value is `false` if this attribute is omitted. If you use JavaScript and Flash on the same page, Java must be running for the `fscommand()` function to work. However, if you use JavaScript only for browser detection or another purpose unrelated to `fscommand()`actions, you can prevent Java from starting by setting `SWLIVECONNECT` to `false`. You can also force Java to start when you are not using JavaScript with Flash by explicitly setting the `SWLIVECONNECT` attribute to `true`. Starting Java substantially increases the time it takes to start a SWF file; set this tag to `true` only when necessary. Applies to the `embed` tag only.

Use the `fscommand()`action to start Java from a stand-alone projector file.

play attribute/parameter

Value

`true` | `false`

Template variable: `$PL`

Description

(Optional) Specifies whether the application begins playing immediately on loading in the browser. If your Flash application is interactive, you might want to let the user initiate play by clicking a button or performing another task. In this case, set the `play` attribute to `false` to prevent the application from starting automatically. The default value is `true` if this attribute is omitted.

loop attribute/parameter

Value

`true` | `false`

Template variable: `$LO`

Description

(Optional) Specifies whether the Flash content repeats indefinitely or stops when it reaches the last frame. The default value is `true` if this attribute is omitted.

quality attribute/parameter

Value

`low` | `medium` | `high` | `autolow` | `autohigh` | `best`

Template variable: `$QU`

Description

(Optional) Specifies the level of anti-aliasing to be used when your application plays. Because anti-aliasing requires a faster processor to smooth each frame of the SWF file before it is rendered on the viewer's screen, select one of the following values based on whether your priority is speed or appearance:

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 frame rate drops below the specified frame rate, anti-aliasing is turned off to improve playback speed. Use this setting to emulate the Antialias command in Flash (View > Preview Mode > Antialias).

Medium applies some anti-aliasing and does not smooth bitmaps. It produces a better quality than the Low setting but a lower quality than the High setting.

High favors appearance over playback speed and always applies anti-aliasing. If the SWF file does not contain animation, bitmaps are smoothed; if the SWF file 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.

The default value for `quality` is `high` if this attribute is omitted.

bgcolor attribute/parameter

Value

#RRGGBB (hexadecimal RGB value)

Template variable: `$BG`

Description

(Optional) Specifies the background color of the application. Use this attribute to override the background color setting specified in the Flash SWF file. This attribute does not affect the background color of the HTML page.

scale attribute/parameter

Value

`showall | noborder | exactfit`

Template variable: `$SC`

Description

(Optional) Defines how the application is placed within the browser window when `width` and `height` values are percentages.

Showall (Default) makes the entire Flash content visible in the specified area without distortion while maintaining the original aspect ratio of the. Borders can appear on two sides of the application.

Noborder scales the Flash content to fill the specified area, without distortion but possibly with some cropping, while maintaining the original aspect ratio of the application.

Exactfit makes the entire Flash content visible in the specified area without trying to preserve the original aspect ratio. Distortion can occur.

The default value is `showall` if this attribute is omitted (and `width` and `height` values are percentages).

align attribute

Value

`Default | L | R | T | B`

Template variable: `$HA`

Description

Specifies the `align` value for the `object`, `embed`, and `img` tags and determines how the Flash SWF file is positioned within the browser window.

Default centers the application in the browser window and crops edges if the browser window is smaller than the application.

L, R, T, and **B** align the application along the left, right, top, and bottom edge, respectively, of the browser window and crop the remaining three sides as needed.

salign parameter

Value

`L | R | T | B | TL | TR | BL | BR`

Template variable: `$SA`

Description

(Optional) Specifies where a scaled Flash SWF file is positioned within the area defined by the `width` and `height` settings. For more information about these conditions, see "scale attribute/parameter" on page 339.

L, **R**, **T**, and **B** align the application along the left, right, top or bottom edge, respectively, of the browser window and crop the remaining three sides as needed.

TL and **TR** align the application to the top left and top right corner, respectively, of the browser window and crop the bottom and remaining right or left side as needed.

BL and **BR** align the application to the bottom left and bottom right corner, respectively, of the browser window and crop the top and remaining right or left side as needed.

If this attribute is omitted, the Flash content is centered in the browser window.

base attribute

Value

base directory or URL

Description

(Optional) Specifies the base directory or URL used to resolve all relative path statements in the Flash SWF file. This attribute is helpful when your SWF files are kept in a different folder from your other files.

menu attribute/parameter

Value

`true` | `false`

Template variable: `$ME`

Description

(Optional) Specifies what type of menu appears when the viewer right-clicks (Windows) or Command-clicks (Macintosh) the application area in the browser.

true shows the full menu, which gives the user several options to enhance or control playback.

false shows a menu that contains only the About Macromedia Flash Player 6 option and the Settings option.

The default value is `true` if this attribute is omitted.

wmode attribute/parameter

Value

`Window` | `Opaque` | `Transparent`

Template variable: `$WM`

Description

(Optional) Lets you use the transparent Flash content, absolute positioning, and layering capabilities available in Internet Explorer 4.0. This attribute/parameter works only in Windows with the Flash Player ActiveX control.

Window plays the application in its own rectangular window on a web page. Window indicates that the Flash application has no interaction with HTML layers and is always the topmost item.

Opaque makes the application hide everything behind it on the page.

Transparent makes the background of the HTML page show through all the transparent portions of the application and can slow animation performance.

Opaque windowless and **Transparent windowless** both interact with HTML layers, letting layers above the SWF file block out the application. The difference between the two is that Transparent allows transparency so that HTML layers below the SWF file might show through if a section of the SWF file has transparency.

The default value is `Window` if this attribute is omitted. Applies to `object` only.

allowscriptaccess attribute/parameter

Value

`always` | `never` | `samedomain`

Description

Use `allowscriptaccess` to let your Flash application communicate with the HTML page hosting it. This is required because `fscommand()` and `getURL()` operations can cause JavaScript to use the permissions of the HTML page, which can be different from the permissions of your Flash application. This has important implications for cross-domain security.

always permits scripting operations at all times.

never forbids all scripting operations.

samedomain permits scripting operations only if the Flash application is from the same domain as the HTML page.

The default value used by all HTML publish templates is `samedomain`.

SeamlessTabbing parameter

Value

`true` | `false`

Description

(Optional) Lets you set the ActiveX control to perform seamless tabbing, so that the user can tab out of a Flash application. This parameter works only in Windows with the Flash Player ActiveX control, version 7 and higher.

true (or omitted) sets the ActiveX control will perform seamless tabbing: after users tab through the Flash application, the next tab keypress will move the focus out of the Flash application and into the surrounding HTML content or to the browser status bar if there is nothing focusable in the HTML following the Flash application.

false sets the ActiveX control to behave as it did in version 6 and earlier: After users tab through the Flash application, the next tab keypress will wrap the focus around to the beginning of the Flash application. In this mode, the focus cannot be advanced past the Flash application by using the tab key.

Previewing the publishing format and settings

To preview your Flash SWF file with your specified publishing format and settings, you can use the Publish Preview command. This command exports the file and opens the preview in the default browser. If you preview a QuickTime video, Publish Preview starts the QuickTime video Player. If you preview a projector, Flash starts the projector.

To preview a file with the Publish Preview command:

1. Define the file's export options using the Publish Settings command; see "Publishing Flash documents" on page 311.
2. Select File > Publish Preview, and from the submenu, select the file format you want to preview.

   Using the current Publish Settings values, Flash creates a file of the specified type(s) in the same location as the FLA. This file remains in this location until you overwrite or delete it.

Using Flash Player

Flash Player plays Flash content in the same way as it appears in a web browser or an ActiveX host application. The player is installed with Flash application. When you double-click Flash content, the operating system starts Flash Player, which then plays the SWF file. You can use the player to make Flash content viewable for users who aren't using a web browser or an ActiveX host application.

You can control Flash content in Flash Player using menu commands and the `fscommand()` function. For example, to make Flash Player take over the entire screen, you assign `fscommand()` to a frame or button and select the `fullscreen` command with the `true` parameter. For more information, see "Sending messages to and from Flash Player" in *Using ActionScript in Flash*.

You can also print Flash content frames using the Flash Player context menu. For more information, see "Printing from the Flash Player context menu" on page 385.

To control applications from Flash Player:

- Do one of the following:
  - Open a new or existing file by selecting File > New or File > Open.
  - Change your view of the application by selecting View > Magnification, and from the submenu, select Show All, Zoom In, Zoom Out, or 100%.
  - Control Flash content playback by selecting Control > Play, Rewind, or Loop.

About configuring a web server for Flash

When your files are accessed from a web server, the server must properly identify them as Flash content to display them. If the MIME type is missing or not properly delivered by the server, the browser can show error messages or a blank window with a puzzle piece icon.

If your server is not properly configured, you (or your server's administrator) must add the Flash SWF file MIME types to the server's configuration files and associate the following MIME types with the SWF file extensions:

- MIME type application/x-shockwave-flash has the .swf file extension.
- MIME type application/futuresplash has the .spl file extension.

If you are administering a server, consult your server software documentation for instructions on adding or configuring MIME types. If you are not administering a server, contact your Internet service provider, webmaster, or server administrator to add the MIME type information.

If your site is on a Macintosh server, you must also set the following parameters: Action: Binary; Type: SWFL; and Creator: SWF2.

CHAPTER 16
Exporting

The Export Movie command in Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 lets you create content that can be edited in other applications and export Flash content directly into a single format. For example, you can export an entire document as a Flash SWF file, as a series of bitmap images, as a single frame or image file, or as moving and still images in various formats, including GIF, JPEG, PNG, BMP, PICT, QuickTime, or AVI.

When you export a Flash file in the SWF format, text is encoded as Unicode, providing support for international character sets, including double-byte fonts. Macromedia Flash Player 6 and later versions support Unicode encoding. For more information, see Chapter 13, "Creating Multilanguage Text," on page 235.

If you have Macromedia Dreamweaver, you can add Flash content to your website easily. Dreamweaver generates all the needed HTML code. You can start Flash from within Dreamweaver to update the Flash content. See "Updating Flash content for Dreamweaver UltraDev" on page 352.

This chapter contains the following sections:

Exporting Flash content and images

To prepare Flash content for use in other applications or to export the contents of the current Flash document in a particular file format, you use the Export Movie and Export Image commands. The Export commands do not store export settings separately with each file, as does the Publish command. (You use the Publish command to create all the files you need to put Flash content on the web. See "Publishing Flash documents" on page 311.)

The Export Movie command lets you export a Flash document to a still-image format and create a numbered image file for every frame in the document. You can also use Export Movie to export the sound in a document to a WAV file (Windows only).

To export the contents of the current frame or the currently selected image to one of the still-image formats or to a single-frame Flash Player application, you use the Export Image command.

Remember the following considerations:

- When you export a Flash image as a vector-graphic file (in Adobe Illustrator format), you preserve its vector information. You can edit these files in other vector-based drawing programs, but you can't import these images into most page-layout and word-processing programs.
- When you save a Flash image as a bitmap GIF, JPEG, PICT (Macintosh), or BMP (Windows) file, the image loses its vector information and is saved with pixel information only. You can edit Flash images exported as bitmaps in image editors such as Adobe Photoshop, but you can no longer edit them in vector-based drawing programs.

To export a Flash document or image:

1. Open the Flash document you want to export, or if you are exporting an image from the document, select the frame or image in the current document that you want to export.
2. Select File > Export Movie or File > Export Image.
3. Enter a name for the output file.
4. Select the file format from the Format pop-up menu.
5. Click Save.

   If the format you selected requires more information, an Export dialog box appears.
6. Set the export options for the format you selected. See the following section.
7. Click OK, then click Save.

About export file formats

You can export Flash content and images in more than a dozen formats, as shown in the following table. Flash content is exported as sequences, and images are exported as individual files. PNG is the only cross-platform bitmap format that supports transparency (as an alpha channel). Some nonbitmap export formats do not support alpha (transparency) effects or mask layers.

For more information on a specific file format, see the sections listed in the following table:

File type	Extension	Windows	Macintosh
"Adobe Illustrator" on page 347	.ai	✔	✔
"Animated GIF, GIF Sequence, and GIF Image" on page 347	.gif	✔	✔
"Bitmap (BMP)" on page 348	.bmp	✔	
"DXF Sequence and AutoCAD DXF Image" on page 348	.dxf	✔	✔
"Enhanced Metafile (Windows)" on page 348	.emf	✔	
"Encapsulated PostScript (EPS) 3.0 with Preview" on page 348	.eps	✔	✔
"Flash document (SWF)" on page 349	.swf	✔	✔

File type	Extension	Windows	Macintosh
"Macromedia Flash Video (FLV)" on page 349	.flv	✔	✔
"JPEG Sequence and JPEG Image" on page 349	.jpg	✔	✔
"PICT (Macintosh)" on page 349	.pct		✔
"PNG sequence and PNG image" on page 350	.png	✔	✔
"QuickTime" on page 350	.mov	✔	✔
"QuickTime Video (Macintosh)" on page 351	.mov		✔
"WAV audio (Windows)" on page 351	.wav	✔	
"Windows AVI (Windows)" on page 351	.avi	✔	
"Windows Metafile" on page 352	.wmf	✔	

Adobe Illustrator

The Adobe Illustrator format is ideal for exchanging drawings between Flash and other drawing applications such as Macromedia FreeHand. This format supports accurate conversion of curve, line style, and fill information. Flash supports importing and exporting Adobe Illustrator 88, 3, 5, 6, and 8 through 10 formats. (See "Importing Adobe Illustrator, EPS, or PDF files" on page 136.) Flash does not support the Photoshop EPS format or EPS files generated using the Print command.

Versions of the Adobe Illustrator format before 5 do not support gradient fills, and only version 6 supports bitmaps.

The Export Adobe Illustrator dialog box lets you select the Adobe Illustrator version—88, 3.0, 5.0, or 6.0.

You can use the Macromedia Flashwriter plug-in to export files in SWF format from Adobe Illustrator 8. Adobe Illustrator versions 9 and 10 have built-in support for SWF export, so the Macromedia Flashwriter plug-in is not needed.

Animated GIF, GIF Sequence, and GIF Image

The Animated GIF, GIF Sequence, and GIF Image option lets you export files in the GIF format. The settings are the same as those available on the GIF tab in the Publish Settings dialog box, with the following exceptions:

Resolution is set in dots per inch (dpi). You can enter a resolution or click Match Screen to use the screen resolution.

Include lets you select to export the minimum image area or specify the full document size.

Colors lets you set the number of colors that can be used to create the exported image—black-and-white; 4-, 6-, 16-, 32-, 64-, 128-, or 256-color; or Standard Color (the standard 216-color, browser-safe palette).

You can also select to interlace, smooth, make transparent, or dither solid colors. For information on these options, see "Configuring publish settings for Flash Player detection" on page 318.

Animation is available for the Animated GIF export format only and lets you enter the number of repetitions, where 0 repeats endlessly.

Bitmap (BMP)

The Bitmap (BMP) format lets you create bitmap images for use in other applications. The Bitmap Export Options dialog box has the following options:

Dimensions sets the size of the exported bitmap image in pixels. Flash ensures that the size you specify always has the same aspect ratio as your original image.

Resolution sets the resolution of the exported bitmap image in dots per inch (dpi) and has Flash automatically calculate width and height based on the size of your drawing. To set the resolution to match your monitor, select Match Screen.

Color Depth specifies the bit depth of the image. Some Windows applications do not support the newer 32-bit depth for bitmap images; if you have problems using a 32-bit format, use the older 24-bit format.

Smooth applies anti-aliasing to the exported bitmap. Anti-aliasing produces a higher-quality bitmap image, but it can create a halo of gray pixels around an image placed on a colored background. Deselect this option if a halo appears.

DXF Sequence and AutoCAD DXF Image

The DXF Sequence and AutoCad DXF Image 3D format lets you export Flash content as AutoCAD DXF release 10 files, so that they can be brought into a DXF-compatible application for additional editing.

This format has no definable export options.

Enhanced Metafile (Windows)

Enhanced Metafile Format (EMF) is a graphics format available in Windows 95 and Windows NT that saves both vector and bitmap information. EMF supports the curves used in Flash drawings better than the older Windows Metafile format. However, some applications do not yet support this graphics format.

This format has no definable export options.

Encapsulated PostScript (EPS) 3.0 with Preview

You can export the current frame as an EPS 3 file for placement in another application, such as a page layout application. An EPS file can be printed by a PostScript printer. As an option, you can include a bitmap preview with the exported EPS file for applications that can import and print the EPS files (such as Microsoft Word and Adobe PageMaker) but that can't display them onscreen.

Flash has no definable exporting options for EPS files.

Flash document (SWF)

You can export the entire document as a Flash SWF file, to place the Flash content in another application, such as Dreamweaver. You can select the same options for exporting a document as you can for publishing the document. See "Publishing Flash documents" on page 311.

Macromedia Flash Video (FLV)

The Macromedia FLV file format lets you import or export a static video stream with encoded audio. This format is intended for use with communications applications, such as video conferencing and files that contain screen share encoded data exported from the Flash Communication Server.

When you export video clips with streaming audio in FLV format, the audio is compressed using the Streaming Audio settings in the Publish Settings dialog box. For information on audio settings, see "Setting publish options for the Flash SWF file format" on page 312.

Files in the FLV format are compressed with the Sorensen codec. See "About the Sorenson Spark codec" on page 179.

To export a video clip in FLV format:

1. Select the video clip in the Library panel.
2. Select Properties from the Library options menu.
3. In the Embedded Video Properties dialog box, click Export.
4. In the Save As dialog box, enter a name for the exported file. Select a location where it will be saved, and click Save.
5. In the Embedded Video Properties dialog box, click OK to close the dialog box.

JPEG Sequence and JPEG Image

The JPEG export options match the JPEG Publish Settings options with one exception: the Match Screen export option makes the exported image match the size of the Flash content as it appears on your screen. The Match Movie publishing option makes the JPEG image the same size as the Flash content and maintains the aspect ratio of the original image.

For more information, see "Specifying publish settings for JPEG files" on page 322.

PICT (Macintosh)

PICT is the standard graphics format on the Macintosh and can contain bitmap or vector information. Use the Export PICT dialog box to set the following options:

Dimensions sets the size of the exported bitmap image specified in pixels. Flash ensures that the size you specify always has the same aspect ratio as your original image.

Resolution sets the resolution in dpi and has Flash automatically calculate width and height based on the size of your drawing. To set the resolution to match your monitor, select Match Screen. Bitmap PICT images usually look best onscreen with 72-dpi resolution.

Include sets the portion of the document to be exported, either Minimum Image Area or Full Document Size.

Color Depth designates whether the PICT file is object-based or bitmap. Object-based images generally look better when printed, and scaling doesn't affect their appearance. Bitmap PICT images normally look best onscreen and can be manipulated in applications such as Adobe Photoshop. You can also select a variety of color depths with bitmap PICT files.

Include Postscript is available only for an object-based PICT file to include information that optimizes printing on a PostScript printer. This information makes the file larger and may not be recognized by all applications.

Smooth Bitmap is available only for a bitmap PICT images. This option applies anti-aliasing in order to smooth jagged edges of a bitmap image.

PNG sequence and PNG image

The PNG export settings options are similar to the PNG publish settings options (see "Specifying publish settings for PNG files" on page 322), with the following exceptions:

Dimensions sets the size of the exported bitmap image to the number of pixels you enter in the Width and Height fields.

Resolution lets you enter a resolution in dpi. To use the screen resolution and maintain the aspect ratio of your original image, select Match Screen.

Colors is the same as the Bit Depth option in the PNG Publish Settings tab and sets the number of bits per pixel to use in creating the image. For a 256-color image, select 8-bit; for thousands of colors, select 24-bit; for thousands of colors with transparency (32 bits) select 24-bit with Alpha. The higher the bit depth, the larger the file.

Include lets you select to export the minimum image area or specify the full document size.

Filter options match those in the PNG Publish Settings tab.

When exporting a PNG sequence or PNG image, you can also apply other options in the PNG Publish Settings, such as Interlace, Smooth, and Dither Solid Colors.

QuickTime

The QuickTime export option creates an application with a Flash track in the same QuickTime format that is installed on your computer. This export format lets you combine the interactive features of Flash with the multimedia and video features of QuickTime in a single QuickTime 4 movie, which can be viewed by anyone with the QuickTime 4 plug-in.

If you import a video clip (in any format) into a document as an embedded file, you can publish the document as a QuickTime movie. If you have imported a video clip in QuickTime format into a document as a linked file, you can also publish the document as a QuickTime movie.

When you export Flash content as a QuickTime movie, all layers in the Flash document are exported as a single Flash track, unless the Flash document contains an imported QuickTime movie. The imported QuickTime movie remains in QuickTime format in the exported application.

These export options are identical to QuickTime publish options. See "Specifying publish settings for QuickTime videos" on page 325.

QuickTime Video (Macintosh)

The QuickTime Video format converts the Flash document into a sequence of bitmaps embedded in the file's video track. The Flash content is exported as a bitmap image without any interactivity. This format is useful for editing Flash content in a video-editing application.

The Export QuickTime Video dialog box contains the following options:

Dimensions specifies a width and height in pixels for the frames of a QuickTime movie. By default, you can specify only the width or the height, and the other dimension is automatically set to maintain the aspect ratio of your original document.To set both the width and the height, deselect Maintain Aspect Ratio.

Format selects a color depth. Options are black-and-white; 4-, 8-, 16-, or 24-bit color; and 32-bit color with alpha (transparency).

Smooth applies anti-aliasing to the exported QuickTime movie. Anti-aliasing produces a higher-quality bitmap image, but it can cause a halo of gray pixels to appear around images when placed over a colored background. Deselect the option if a halo appears.

Compressor selects a standard QuickTime compressor. For more information, see your QuickTime documentation.

Quality controls the amount of compression applied to your Flash content. The effect depends on the compressor selected.

Sound Format sets the export rate for sounds in the document. Higher rates yield better fidelity and larger files. Lower rates save space.

WAV audio (Windows)

The WAV Export Movie option exports only the sound file of the current document to a single WAV file. You can specify the sound format of the new file.

Select Sound Format to determine the sampling frequency, bit rate, and stereo or mono setting of the exported sound. Select Ignore Event Sounds to exclude events sounds from the exported file.

Windows AVI (Windows)

This format exports a document as a Windows video but discards any interactivity. The standard Windows movie format, Windows AVI, is a good format for opening a Flash animation in a video-editing application. Because AVI is a bitmap-based format, documents that contain long or high-resolution animations can quickly become very large.

The Export Windows AVI dialog box has the following options:

Dimensions specifies a width and height, in pixels, for the frames of an AVI movie. Specify only the width or the height; the other dimension is automatically set to maintain the aspect ratio of your original document. Deselect Maintain Aspect Ratio to set both the width and the height.

Video Format selects a color depth. Some applications do not yet support the Windows 32-bit image format. If you have problems using this format, use the older 24-bit format.

Compress Video displays a dialog box for selecting standard AVI compression options.

Smooth applies anti-aliasing to the exported AVI movie. Anti-aliasing produces a higher-quality bitmap image, but it can cause a halo of gray pixels to appear around images when placed over a colored background. Deselect the option if a halo appears.

Sound Format lets you set the sample rate and size of the sound track, and whether it will be exported in mono or stereo. The smaller the sample rate and size, the smaller the exported file, with a possible trade-off in sound quality. For more information on exporting sound to the AVI format, see "Compressing sounds for export" on page 209.

Windows Metafile

Windows Metafile format is the standard Windows graphics format and is supported by most Windows applications. This format yields good results for importing and exporting files. It has no definable export options. See "Enhanced Metafile (Windows)" on page 348.

Updating Flash content for Dreamweaver UltraDev

If you have Dreamweaver UltraDev installed on your system, you can export Flash SWF files directly to a Dreamweaver UltraDev site. For more information on working with Dreamweaver UltraDev, see *Using Dreamweaver*.

In Dreamweaver UltraDev, you can add the Flash content to your page. With a single click, you can update the Flash document (FLA) and re-export the updated Flash content to UltraDev automatically.

To update Flash content for Dreamweaver UltraDev:

1. In Dreamweaver UltraDev, open the HTML page that contains the Flash content.
2. Do one of the following:
   - Select the Flash content and click Edit in the Property inspector.
   - In Design view, press Control (Windows) or Command (Macintosh), and double-click the Flash content.
   - In Design view, right-click (Windows) or Control-click (Macintosh) the Flash content, and select Edit with Flash from the context menu.
   - In the Site panel, right-click (Windows) or Control-click (Macintosh) the Flash content in Design view, and select Open with Flash from the context menu.

   The Flash application is started on your system.
3. If the Flash file (FLA) for the exported file does not open, a file locator dialog box appears. Navigate to the FLA file in the Open File dialog box, and click Open.
4. If the user has used the Change Link Sitewide feature in Dreamweaver UltraDev, a warning is shown. Click OK to apply link changes to the Flash content. Click Don't Warn Me Again to prevent the warning message from appearing when you update the Flash content.

5. Update the Flash document (FLA) as needed in Flash.
6. To save the Flash document (FLA) and re-export the Flash content to Dreamweaver, do one of the following:
   - To update the file and close Flash, click the Done button above the upper left corner of the Stage.
   - To update the file and keep Flash open, select File > Update for Dreamweaver.

CHAPTER 17
Creating Accessible Content

You can create Flash content that is accessible to all users, including those with disabilities, using the accessibility features provided with Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004. As you design accessible Flash applications, consider how users might interact with the content. For example, visually impaired users might rely on assistive technology, such as screen readers that provide an audio version of screen content, and hearing-impaired users might read text and captions in the document. Other considerations arise for users with mobility or cognitive impairments.

For a quick lesson in how to create accessible content, select Help > How Do I > Quick Tasks > Create Accessible Flash Content. To view a sample of an accessible application, see "Using the accessibility features in Flash" on page 421.

You can create accessible content with Flash by using accessibility features included in the authoring environment user interface, taking advantage of ActionScript designed to implement accessibility, and following recommended design and development practices. The list of recommended practices that follows is not exhaustive, but rather suggests common issues to consider. Depending on your audience's needs, additional requirements might arise.

Visually impaired users For visually impaired users, including those with color blindness, remember the following design recommendations:

- Use the Accessibility panel or ActionScript to provide a description of your document and nontext elements for use with a screen reader. See "Using Flash to enter accessibility information for screen readers" on page 359 and "Creating accessibility with ActionScript" on page 369.
- Describe the layout of your Flash application and the individual controls used to navigate through it. See "Using Flash to enter accessibility information for screen readers" on page 359.
- Design and implement a logical tab order using either the Accessibility panel or ActionScript. See "Creating a tab order index for keyboard navigation in the Accessibility panel (Flash Professional only)" on page 367 and "Using ActionScript to create a tab order for accessible objects" on page 371.
- Design the document so that constant changes in the Flash content do not cause screen readers to refresh unnecessarily. For example, you should group or hide looping elements. See "Hiding an object from the screen reader" on page 364.

- Provide captions for narrative audio. Be aware of audio in your document that might interfere with a user being able to listen to the screen reader. See "Testing accessible content" on page 373.
- Ensure that color is not the only means of conveying information. In addition, make sure that the foreground and background contrast sufficiently to make text readable for people with low vision and color blindness.

Users with visual or mobility impairment For users with either visual or mobility impairment, ensure that controls are device independent (or accessible by keyboard).

Hearing-impaired users For hearing impaired users, you can caption audio content. See "Accessibility for hearing-impaired users" on page 372.

Users with cognitive impairment Users with cognitive impairments often respond best to uncluttered design that is easily navigable.

This chapter contains the following sections:

Worldwide accessibility standards

Many countries, including the United States, Australia, Canada, Japan, and countries in the European Union, have adopted accessibility standards based on those developed by the World Wide Web Consortium (W3C). The W3C publishes the Web Content Accessibility Guidelines, a document that prioritizes actions designers should take to make web content accessible. For information about the Web Accessibility Initiative, see the W3C website at www.w3.org/WAI.

In the United States, the law that governs accessibility is commonly known as Section 508, which is an amendment to the U.S. Rehabilitation Act. Section 508 prohibits federal agencies from buying, developing, maintaining, or using electronic technology that is not accessible to those with disabilities. In addition to mandating standards, Section 508 allows government employees and the public to sue agencies in federal court for noncompliance.

For additional information about Section 508, see the following websites:

- The US government-sponsored website at www.section508.gov
- The Macromedia accessibility site at www.macromedia.com/macromedia/accessibility/

Macromedia Flash Accessibility web page

For the latest information on creating and viewing accessible Flash content, including supported platforms, screen reader compatibility, articles, and accessible examples, consult the Macromedia Flash Accessibility web page at www.macromedia.com/software/Flash/productinfo/accessibility/.

Understanding screen reader technology

Screen readers are software designed to navigate through a website and read the web content aloud. Visually impaired users often rely on this technology. You can create Flash content designed for use with screen readers on only Windows platforms . Viewers of your Flash content must have Flash Player 6 or later and Internet Explorer on Windows 98 or later.

JAWS, from Freedom Scientific, is one example of screen reader software. You can access the JAWS page of the Freedom Scientific website at www.hj.com/fs_products/software_jaws.asp. Another commonly used screen reader program is Window-Eyes, from GW Micro. To access the latest information on Window-Eyes, visit the GW Micro website at www.gwmicro.com. To enable a screen reader to read nontextual objects in your application, such as vector art and animations, you can use the Accessibility panel to associate a name and description with the object, which the screen reader reads aloud.

Screen readers help users understand what is contained in a web page or Flash document. Based on the keyboard shortcuts you define, you can allow users to navigate through your document using the screen reader with ease. See "Creating a keyboard shortcut" on page 364.

To expose graphic objects, you can provide a description using the Accessibility panel or ActionScript. See "Using Flash to enter accessibility information for screen readers" on page 359.

Because different screen reader applications use varying methods to translate information into speech, the presentation of your content will vary according to each user. As you design accessible applications, remember that you have no control over how any screen reader will behave. You have control over only the content, which you can mark up in your Flash applications to expose the text and ensure screen reader users can activate the controls. This means that you can decide which objects in the Flash application are exposed to screen readers, provide descriptions for them, and decide the order in which they are exposed to screen readers. However, you cannot force screen readers to read specific text at specific times or control the manner in which that content is read. It is very important, therefore, to test your applications with a variety of screen readers to ensure that they perform as you expect. See "Testing accessible content" on page 373.

Flash and Microsoft Active Accessibility (Windows only)

Flash Player is optimized for Microsoft Active Accessibility (MSAA) which provides a highly descriptive and standardized way for applications and screen readers to communicate. MSAA is available for Windows operating systems only. For more information on Microsoft Accessibility Technology, visit the Microsoft Accessibility website at www.microsoft.com/enable/default.aspx.

The Windows ActiveX (Internet Explorer plug-in) version of Flash Player 6 supports MSAA, but the Windows Netscape and Windows stand-alone players do not.

Caution: MSAA is currently *not* supported in the opaque windowless and transparent windowless modes. (These modes are options in the HTML Publish Settings panel, available for use with the Windows version of Internet Explorer 4.0 or later, with the Flash ActiveX control.) If you need your Flash content to be accessible to screen readers, avoid using these modes.

Flash Player makes information about the following types of accessibility objects available to screen readers using MSAA. To understand how to enter accessible information for each object, see "Using Flash to enter accessibility information for screen readers" on page 359.

Dynamic or static text The principal property of a text object is its name. To comply with MSAA convention, the name is equal to the contents of the text string. A text object can also have an associated description string. Flash uses the static or dynamic text immediately above or to the left of an input text field as a label for that field.

Note: Any text that is a label is *not* passed to a screen reader. Instead, the content of that text is used as the name of the object that it labels. Labels are never assigned to buttons or text fields that have author-supplied names.

Input text fields An input text object has a value, an optional name, a description string, and a keyboard shortcut string. As with dynamic text, an input text object's name can come from a text object that is above or to the left of it.

Buttons A button object has a state (pressed or not pressed), supports a programmatic default action that causes the button to depress momentarily, and can optionally have a name, a description string, and a keyboard shortcut string. As with text input fields, for buttons, Flash uses any text entirely inside a button as a label for that button.

Note: For accessibility purposes, movie clips used as buttons with button event handlers such as `onPress` are considered buttons—not movie clips—by Flash Player.

Components Flash UI components provide special accessibility implementation. For more information, see "Using accessible components" on page 369 and "Creating accessibility with ActionScript" on page 369.

Movie clips Movie clips are exposed to screen readers as graphic objects when they do not contain any other accessible objects, or when the Accessibility panel is used to provide a name or a description for a movie clip. When a movie clip contains other accessible objects, the clip itself is ignored, and the objects inside it are made available to screen readers.

Note: All Flash Video objects are treated as simple movie clips.

Basic accessibility support in Flash Player

Flash Player provides some basic accessibility support for all Flash documents, whether or not they are designed using the accessibility features found in the Flash authoring tool. This generic support for documents that do not use any accessibility features includes the following:

Dynamic or static text Text is transferred to the screen reader program as a name, but with no description.

Input text Text is transferred to the screen reader. No names are transferred, except where labeling relationships are found, and no descriptions or keyboard shortcut strings are transferred.

Buttons The state of the button is transferred to the screen reader. No names are transferred, except where labeling relationships are found, and no descriptions or keyboard shortcut strings are transferred.

Documents The document state is transferred to the screen reader, but with no name or description.

Using Flash to enter accessibility information for screen readers

Screen readers read aloud a description of the content, read text, and assist users as they navigate through the user interfaces of traditional applications such as menus, toolbars, dialog boxes, and input text fields.

By default, the following objects are defined as accessible in all Flash documents and are included in the information that Flash Player provides to screen reader software:

- Dynamic text
- Input text fields
- Buttons
- Movie clips
- Entire Flash applications

Flash Player automatically provides names for static and dynamic text objects, which are simply the contents of the text. For each of these accessible objects, you can set descriptive properties for screen readers to read aloud. You can also control how Flash Player decides which objects to expose to screen readers—for example, you can specify that certain accessible objects are not exposed to screen readers at all.

The Flash Accessibility panel

One way to provide accessibility information to screen readers is to use the Flash Accessibility panel. The alternate approach is to enter accessibility information using ActionScript. See "Creating accessibility with ActionScript" on page 369.

The Accessibility panel is a self-contained property inspector that lets you set accessibility options for individual Flash objects or entire Flash applications.

If you select an object on the Stage, you can make that object accessible and then specify options such as a name, description, keyboard shortcut, and tab index order (Flash Professional only) for the object. For movie clips, you can specify whether child object information is passed to the screen reader (this option is selected by default when you make an object accessible).

With no objects selected on the Stage, you use the Accessibility panel to assign accessibility options for an entire Flash application. You can make the entire application accessible, make child objects accessible, have Flash label objects automatically, and give specific names and descriptions to objects.

All objects in Flash documents must have instance names for you to apply accessibility options to them. You create instance names for objects in the Property inspector. The instance name is used to refer to the object in ActionScript.

To open the Accessibility panel:

1. Select Window > Other Panels > Accessibility.
2. Select from the available options:

Make Object Accessible instructs Flash Player to pass the accessibility information for an object to a screen reader. This option is selected by default; when the option is disabled, accessibility information for the object is not passed to screen readers. You might find it useful to disable this option as you test content for accessibility because some objects may be extraneous or decorative and making them accessible could produce confusing results in the Screen Reader. You can then apply a name manually to the labeled object, and hide the labeling text by deselecting Make Object Accessible. When Make Object Accessible is disabled, all other controls on the Accessibility panel are disabled.

Make Child Objects Accessible instructs Flash Player to pass child object information to the screen reader. This option is for movie clips only and is selected by default. Disabling this option for a movie clip causes that movie clip to appear as a simple clip in the accessible object tree, even if the clip contains text, buttons, and other objects. All objects within the movie clip are then hidden from the object tree. Like the Make Object Accessible option, this option is useful mainly for hiding extraneous objects from screen readers.

Note: If a movie clip is used as a button, meaning that it has a button event handler assigned to it, such as `onPress` or `onRelease`, the Make Child Objects Accessible option is ignored because buttons are always treated as simple clips, and their children are never examined, except in the case of labels.

Auto Label instructs Flash to automatically label objects on the Stage with the text associated with them. See "Using automatic labeling" on page 361.

Name specifies the object name. Screen readers identify objects by reading these names aloud. When accessible objects don't have specified names, a screen reader might read a generic word, such as *Button*, which can be confusing.

Caution: Do not confuse object names specified in the Accessibility panel with instance names specified in the Property inspector.

Description lets you enter a description of the object to the screen reader. This description is read by the screen reader.

Shortcut is used to describe keyboard shortcuts to the user. The text entered in this text box is read by the screen reader. Entering keyboard shortcut text here does not create a keyboard shortcut for the selected object. You must provide ActionScript keyboard handlers in order to create shortcut keys. For more information, see "Creating a keyboard shortcut" on page 364.

Tab Index (Flash Professional only) creates a tab order in which objects are accessed when the user presses the tab key. The tab index feature works for keyboard navigation through a page, but not for screen reader reading order. For information on how to use this field, see "Creating a tab order index for keyboard navigation in the Accessibility panel (Flash Professional only)" on page 367.

For more information, see the Macromedia Flash Accessibility web page at www.macromedia.com/software/Flash/productinfo/accessibility/.

Selecting names for buttons, text fields, and entire Flash applications

You can use the Accessibility panel to assign names to buttons and input text fields so that they are identified appropriately by the screen reader. There are two ways of doing this:

- Use the auto label feature to assign text adjacent or within the object as a label.
- Enter a specific label in the Accessibility panel name field.

Using automatic labeling

Flash automatically gives an appropriate name to a button or input text field in your document, as a text label that you have placed on top of, inside, or near a button or another text field. Labels for buttons must appear within the bounding shape of the button. For the button in the following illustration, most screen readers would first read the word *button*, then read the text label *Home*. The user can press Return or Enter to activate the button.

A form might include an input text field where users enter their names. A static text field, with the text *Name* appears next to the input text field. When Flash Player discovers such an arrangement, it assumes that the static text object is a serving as a label for the input text field.

For example, when the following part of a form is encountered, a screen reader reads "Enter your name here."

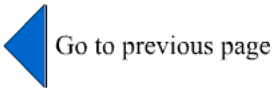
In the Accessibility panel, you can turn off automatic labeling if it is not appropriate for your document. You can also turn off automatic labeling for specific objects within your document. See "Turning off automatic labeling for an object and specifying a name" on page 363.

Providing a name for an object

If you do not want to use automatic labeling for the entire application, you can turn it off and provide names for the objects in the Accessibility panel. If you have automatic labeling turned on, you can also select specific objects and provide names for the objects in the Name text box in the Accessibility panel so that the name is used instead of the object text label.

When a button or input text field doesn't have a text label, or when the label is in a location that Flash Player can't detect, you can specify a name for the button or text field. You can also specify a name if the text label is near a button or text field, but you don't want that text to be used as that object's name.

For example, in the following figure, the text that describes the button appears outside and to the right of the button. In this location, Flash Player does not detect the text, and it is not read by the screen reader.



To rectify this, open the Accessibility panel and select the button and enter the desired name (for example, “left arrow”) and description (for example, “Go to previous page”) in the Name and Description text boxes, respectively. To prevent repetition, make the text object inaccessible.

Note: An object’s accessibility name is unrelated to the ActionScript instance name or ActionScript variable name associated with the object. For information on how ActionScript handles instance names and variable names in text fields, see “About text field instance and variable names” in *Using ActionScript in Flash*. (This information generally applies to all objects.)

To specify a name and description for a button, text field, or entire Flash application:

1. Do one of the following:
   - To provide a name for a button or text field, select the object on the Stage.
   - To provide a name for an entire Flash application, deselect all objects on the Stage.
2. Do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.
3. In the Accessibility panel, make sure that the Make Object Accessible (for buttons or text fields) or Make Movie Accessible (for entire Flash applications) option is selected (the default setting).
4. Enter a name for the button, text field, or Flash application in the Name text box.
5. Enter a description for the button, text field, or Flash application in the Description text box.

To define accessibility for a selected object in a Flash application:

1. Select the object on the Stage and do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.
2. In the Accessibility panel, do one of the following:
   - Select Make Object Accessible (the default setting) to expose the object to screen readers and to enable other options in the panel.
   - Deselect Make Object Accessible to hide the object from screen readers. This disables the other options in the panel.

3. Enter information for the selected object as needed:

Dynamic text Enter a name for the text object in the Name text box and an optional description of the text in the Description text box. (To provide a description for static text, you must convert it to dynamic text.)

Input text fields or buttons Enter a name for the object. Enter a description of the object in the Description text box. Enter a keyboard shortcut in the Shortcut text box.

Movie clips Enter a name for the object. Enter a description in the Description text box. Select Make Child Objects Accessible to expose the objects inside the movie clip to screen readers.

Note: If your application can be described in a simple phrase of text that can be easily conveyed by a screen reader, turn off the Make Children Accessible option for your document, and type in a suitable description.

Specifying advanced accessibility options for screen readers

Flash provides several accessibility authoring features that go beyond simply providing names for objects. In addition to providing descriptions for text or text fields, buttons, or movie clips, and keyboard shortcuts for input text fields or buttons, you can also turn off automatic labeling behavior for your document.

You can also hide a selected object from screen readers. For example, you should hide objects that are repetitive or do not convey information. You can also decide to hide accessible objects that are contained inside a movie clip or Flash application and expose only the movie clip or Flash application to screen readers.

Turning off automatic labeling for an object and specifying a name

You can specify a name for an individual object if automatic labeling does not provide the correct information.

To turn off an automatic label for an individual object and specify a name for it:

1. On the Stage, select the button or input text field for which you want to control labeling.
2. Do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.
3. In the Accessibility panel, select Make Object Accessible (the default setting).
4. Enter a name for the object in the Name text box.

   The name is read as the label for the button or text field.
5. To turn off accessibility for the automatic label (and hide it from screen readers), select the text object on the Stage.
6. If the text object is static text, convert it to dynamic text (in the Property inspector, select Dynamic Text from the Text type pop-up menu).
7. In the Accessibility panel, deselect Make Object Accessible.

Hiding an object from the screen reader

You can hide an object from the screen reader simply by turning off accessibility for the object. You should only hide objects that are repetitive or convey no content. When an object is hidden, the screen reader ignores the object.

1. On the Stage, select the button or input text field you want to hide from the screen reader.
2. Do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.
3. In the Accessibility panel, do one of the following:
   - If the object is a movie clip, button, text field, or another object, deselect Make Object Accessible.
   - If the object is the child of a movie clip, deselect Make Child Objects Accessible.

Creating a keyboard shortcut

You can create a keyboard shortcut for an object, such as a button, so users can quickly navigate to it without listening to the contents of an entire page. For example, you can create a keyboard shortcut so users can quickly navigate to a menu, a toolbar, the next page, or a submit button.

Two steps are required to create a keyboard shortcut:

- Code the ActionScript to create a keyboard shortcut for an object. See "`Key class`" in *Flash ActionScript Language Reference*. If you provide a keyboard shortcut for an input text field or button, you must also use the ActionScript Key class to detect the key the user presses during Flash content playback. See "Capturing keypresses" in *Using ActionScript in Flash*.
- Select the object and add the name of the keyboard shortcut to the Accessibility panel so the screen reader can read it.

Keyboard shortcut functionality also depends on the screen reader software used. Make sure to test your Flash content with multiple screen readers. The key combination Control+F, for example, is a reserved keystroke for both the browser and the screen reader. The arrow keys are also reserved by the screen reader. Generally, you can use the keys 0-9 on the keyboard for keyboard shortcuts. However, even these keys are increasingly used by screen readers, so it is very important to test your keyboard shortcuts. See "Testing accessible content" on page 373.

To indicate the name of a keyboard shortcut for the screen reader:

1. On the Stage, select the button or input text field for which you want to create a keyboard shortcut.
2. Do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.

3. In the Shortcut field, type the name of the keyboard shortcut, using the following conventions:
   - Spell out key names, such as Control or Alt.
   - Use capital letters for alphabetic characters.
   - Use a plus sign (+) between key names, with no spaces (for example, Control+A).

   Warning: Flash does not check that the ActionScript to code the keyboard shortcut has been created.

Keyboard shortcut example

To create a keyboard shortcut, Control+7, for a button with the instance name myButton, you would do the following:

1. Select the object on the Stage, display the Accessibility panel, and in the Shortcut field, type **Control+7**.
2. Enter the following code in the Actions panel:

```
function myOnPress() {
  trace( "hello" );
}
function myOnKeyDown() {
  if (Key.isDown(Key.CONTROL) && Key.getCode() == 55) // 55 is key code for
  7
  {
    Selection.setFocus( myButton );
    myButton.onPress();
  }
}
var myListener = new Object();
myListener.onKeyDown = myOnKeyDown;
Key.addListener( myListener );
myButton.onPress = myOnPress;
myButton._accProps.shortcut = "Ctrl+7"
Accessibility.updateProperties();
```

   Note: The example assigns the keyboard shortcut Control+7 to a button with an instance name of myButton and makes information about the shortcut available to screen readers. In this example, when you press Control+7 the `myOnPress` function displays the text "hello" in the Output panel. See `Key.addListener()` in *Flash ActionScript Language Reference*.

Making an entire Flash application accessible

After a Flash document is complete and ready to be published, make the entire Flash application accessible.

To define accessibility for an entire Flash application:

1. When the Flash document is complete and ready to be published or exported, deselect all elements in the document and do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.

2. In the Accessibility panel, select Make Movie Accessible (the default setting) to expose the document to screen readers.
3. Select or deselect the Make Children Accessible option to expose or omit any accessible objects in the document to screen readers.
4. If you selected Make Movie Accessible in step 3, enter information for the document as needed:
   - Enter a name for the document in the Name text box.
   - Enter a description of the document in the Description text box.
5. Select Auto Label (the default setting) to use text objects as automatic labels for accessible buttons or input text fields contained in the document. Deselect this option to turn off automatic labeling and expose text objects to screen readers as text objects.

Using sound with screen readers

Sound is the most important medium for most screen reader users. Consider how any sound in your document will interact with the text spoken aloud by screen readers. It might be difficult for screen reader users to hear what their screen readers are saying if your Flash application contains loud sounds.

Viewing and creating tab order and reading order

There are two aspects to tab indexing order—the *tab order* in which a user navigates through the web content and the order in which things are read by the screen reader, called the *reading order*.

Flash Player uses a tab index order from left to right and top to bottom. However, if this is not the order you want to use, you can customize both the tab and reading order using the `tabIndex` property in ActionScript (In ActionScript, the `tabIndex` property is synonymous with the reading order).

Tab order You can create a tab order that determines the order in which objects receive input focus when users press the Tab key. You can use ActionScript to do this, or if you have Flash MX 2004 Professional, you can use the Accessibility panel to specify the tab order. Remember that the tab index that you assign in the Accessibility panel does not necessarily control the reading order. See "Creating a tab order index for keyboard navigation in the Accessibility panel (Flash Professional only)" on page 367.

Reading order You can also control the order in which a screen reader reads information about the object (known as the reading order). To create a reading order, you must use ActionScript to assign a tab index to every instance. You must create a tab index for every accessible object, not just the focusable objects. For example, dynamic text must have tab indexes, even though a user cannot tab to dynamic text. If you do not produce a tab index for every accessible object in a given frame, Flash Player ignores all tab indexes for that frame whenever a screen reader is present, and uses the default tab ordering instead. See "Using ActionScript to create a tab order for accessible objects" on page 371.

Creating a tab order index for keyboard navigation in the Accessibility panel (Flash Professional only)

You can create a tab order index in the Accessibility panel for keyboard navigation. You can create a custom tab order for the following objects:

- Dynamic text
- Input text
- Buttons
- Movie clips, including compiled movie clips
- Components
- Screens

Note: You can also use ActionScript to create a keyboard navigation tab order index. See "Using ActionScript to create a tab order for accessible objects" on page 371.

Tab focus occurs in numerical order, starting from the lowest index number. Once tab focus reaches the highest tab index, focus returns to the lowest index number.

When you move user-set tab indexed objects around in your document, or to another document, Flash retains the index attributes. You should then check for and resolve index conflicts (for example, two different objects on the Stage that have the same tab index number).

Caution: If two or more objects have the same tab index in any given frame, Flash follows the order in which the objects were placed on the Stage. Therefore, you should resolve all tab index conflicts to be sure the desired tab order index is achieved.

To create a tab order index:

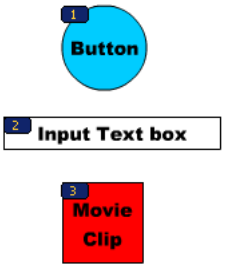
1. Select the object in which to assign a tab order and do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.
2. If you're providing an index for the selected object only, in the Tab Index text box, enter a positive integer (up to 65535) that reflects the order in which the selected object should receive focus.

   Note: For information about creating a tab order using ActionScript, see "Using ActionScript to create a tab order for accessible objects" on page 371. Tab indexes created in ActionScript do not appear on Stage when the Show Tab Order option is enabled.

To view a tab order:

- Select View > Show Tab Order.

  Tab index numbers for individual objects appear in the upper left corner of the object.



Note: Tab order created with ActionScript code, rather than the Accessibility panel, does not appear when you enable the Show Tab Order option.

About animation and accessibility for the visually impaired

In some situations, you might want to change the property of an accessible object during movie playback. For example, you might want to indicate changes that take place on a keyframe in an animation.

To update properties for an accessible object:

1. Display the frame in which you want to change the properties.
2. Do one of the following:
   - Select Window > Properties if the inspector is not visible. In the Property inspector, click the Accessibility button.
   - Select Window > Other Panels > Accessibility.
3. In the Accessibility panel, change the properties for the object as needed.

Alternatively, you can use ActionScript to update accessibility properties. See "Creating accessibility with ActionScript" on page 369.

Different screen readers treat new objects on frames differently. Some screen readers may read only the new object; some screen readers may re-read the entire document.

Try to avoid animating the text, buttons, and input text fields in your document. If you keep these kinds of objects stable, you reduce the chance of causing a screen reader to emit extra "chatter" that can annoy users. Also, it's best to avoid making your Flash content loop.

If you're using a feature such as Text Break Apart to animate text, Flash Player can't determine the actual text content of that text. Other examples of information-carrying graphics include icons and gestural animations. You can remedy problems such as this by providing names or descriptions for certain accessible objects within your document or for the entire Flash application. See "Making an entire Flash application accessible" on page 365. You can also add supplementary text into your document or shift important content from graphics to text.

Using accessible components

To accelerate building accessible applications, Macromedia has built a core set of UI components. These components automate many of the most common accessibility practices related to labeling, keyboard access, and testing and help ensure a consistent user experience across rich applications. Flash comes with the following set of accessible components:

- SimpleButton
- CheckBox
- RadioButton
- Label
- TextInput
- TextArea
- ComboBox
- ListBox
- Window
- Alert
- DataGrid

Accessible Flash components have special requirements to work with screen readers. The components must contain ActionScript that defines their accessible behavior. For information on which accessible components work with screen readers, see the Macromedia Flash Accessibility web page at www.macromedia.com/software/Flash/productinfo/accessibility/.

For general information about components, see Chapter 1, "About Components" in *Using Components*.

For each accessible component, you enable the accessible portion of the component with the `enableAccessibility()` command. This command includes the accessibility object with the component as the document is compiled. Because there is no simple way to remove an object after it has been added to the component, these options are disabled by default. Therefore, it's important that you enable accessibility for each component. This step needs to be done only once for each component; it is not necessary to enable accessibility for each instance of a component for a given document. See "Button component", "CheckBox component", "ComboBox component", "Label component", "List component", "RadioButton component", and "Window component" in *Using Components*.

Creating accessibility with ActionScript

In addition to the accessibility features included in the Flash user interface, you can create accessible documents with ActionScript. For accessibility properties that apply to the entire document, you can create or modify a global variable called `_accProps`. See `_accProps` in *Flash ActionScript Language Reference*.

For properties that apply to a specific object, you can use the syntax `instancename._accProps`. The value of `_accProps` is an object that can include any of the following properties:

Property	Type	Equivalent selection in the Accessibility panel	Applies to
`.silent`	Boolean	Make Movie Accessible/Make Object Accessible (inverse logic)	Entire documents Buttons Movie clips Dynamic text Input text
`.forceSimple`	Boolean	Make Child Objects Accessible (inverse logic)	Entire documents Movie clips
`.name`	string	Name	Entire documents Buttons Movie clips Input text
`.description`	string	Description	Entire documents Buttons Movie clips Dynamic text Input text
`.shortcut`	string	Shortcut	Buttons Movie clips Input text

Note: Inverse logic means that a value of `true` in ActionScript corresponds to a checkbox that is not selected in the Accessibility panel, and a value of `false` in ActionScript corresponds to a selected check box in the Accessibility panel.

Modifying the `_accProps` variable has no effect by itself. You must also use the `Accessibility.updateProperties` method to inform screen reader users of Flash content changes. Calling the method causes Flash Player to re-examine all accessibility properties, update property descriptions for the screen reader, and, if necessary, send events to the screen reader that indicate changes have occurred.

When updating accessibility properties of multiple objects at once, you need to include only a single call to `Accessiblity.updateProperties` (too frequent updates to the screen reader can cause some screen readers to become too verbose).

See `Accessibility.updateProperties()` in *Flash ActionScript Language Reference*.

Implementing screen reader detection with the Accessibility.isActive() method

To create Flash content that behaves in a specific way if a screen reader is active, you can use the ActionScript method `Accessibility.isActive()`, which returns a value of `true` if a screen reader is present, and `false` otherwise. You can then design your Flash content to perform so that it's compatible with screen reader use (for example, by hiding child elements from the screen reader). For more information, see `Accessibility.isActive()` in *Flash ActionScript Language Reference*.

For example, you could use the `Accessibility.isActive()` method to decide whether to include unsolicited animation. Unsolicited animation happens without the screen reader doing anything, which can be confusing for screen readers.

The `Accessibility.isActive()` method provides asynchronous communication between the Flash content and Flash Player, which means that a slight real-time delay can occur between the time the method is called and the time in which Flash Player becomes active, returning an incorrect value of `false`. To ensure that the method is called correctly, you can do one of the following:

- Instead of using the `Accessibility.isActive()` method when the Flash content first plays, call the method whenever you need to make a decision about accessibility.
- Introduce a short delay of one or two seconds at the beginning of your document to give the Flash content enough time to contact Flash Player.

  For example, you can attach this method with an `onFocus` event to a button. This generally gives the SWF file enough time to load and you can safely assume a screen reader user will tab to the first button or object on the Stage.

Using ActionScript to create a tab order for accessible objects

In addition to assigning a tab index to objects with the Accessibility panel (see "Creating a tab order index for keyboard navigation in the Accessibility panel (Flash Professional only)" on page 367), you can create the tab order with ActionScript by assigning the `tabIndex` property to the following objects:

- Dynamic text
- Input text
- Buttons
- Movie clips, including compiled movie clips
- Timeline frames
- Screens

If you create a tab order for a frame and you don't specify a tab order for an accessible object in the frame, Flash Player ignores all the custom tab order assignments. You should, therefore, provide a complete tab order for all accessible objects. Additionally, all objects assigned to a tab order, except frames, must have an instance name specified in the Instance Name text box of the Property inspector. Even items that are not tab stops, such as text, need to be included in the tab order if they are to be read in that order.

Because static text cannot be assigned an instance name, it cannot be included in the list of the `tabIndex` property values. As a result, a single instance of static text anywhere in the SWF file causes the reading order to revert to the default.

To specify a tab order, you assign an order number to the `tabIndex` property, as shown in the following example:

```
_this.myOption1.btn.tabIndex = 1
_this.myOption2.txt.tabIndex = 2
```

See `Button.tabIndex`, `MovieClip.tabIndex`, and `TextField.tabIndex` in *Flash ActionScript Language Reference*.

You can also use `tabChildren()` or `tabEnabled()` methods to assign custom tab order. See `MovieClip.tabChildren`, `MovieClip.tabEnabled`, and `TextField.tabEnabled` in *Flash ActionScript Language Reference*.

Accessibility for hearing-impaired users

To provide accessibility for hearing-impaired users, you can include captions for audio content that is integral to comprehending the material. A video of a speech, for example, would probably require captions for accessibility, but a quick sound associated with a button probably wouldn't.

There are several ways you can add captions to a Flash document, including the following:

- Add text as captions, taking care to ensure the captions are synchronized on the Timeline with the audio.
- Use Hi-Caption Viewer, a component available from Hi Software that works with Hi-Caption SE for use with Flash. The white paper titled *Captioning Multimedia with Hi-Caption SE for Use with Macromedia Flash MX* explains how to use Hi-Caption SE and Flash together to create an a captioned document. The white paper is available on the Macromedia website on the Accessibility White Papers page at www.macromedia.com/macromedia/accessibility/whitepapers/. For more information on Hi-Caption SE, see the link on the Macromedia Accessibility Captioning page at www.macromedia.com/macromedia/accessibility/tools/caption.html.

Testing accessible content

When you test your accessible Flash applications, follow these recommendations:

- If you are designing your document to work with screen readers, download several screen readers and test your application by playing it in a browser with the screen reader enabled. Make sure that the screen reader is not attempting to "talk over" places in your document where you have inserted separate audio. Several screen reader applications provide a demonstration version of the software as a free download; you should try as many as you can to ensure compatibility across screen readers.
- If you are creating interactive content, test it and verify that users can navigate your content effectively using only the keyboard. This can be an especially challenging requirement because different screen readers work in different ways when processing input from the keyboard—meaning that your Flash content might not receive keystrokes as you intended. Make sure to test all keyboard shortcuts.

CHAPTER 18
Printing from SWF Files

You can add printing functionality to your Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 document that lets users print from Flash Player. You can use the ActionScript PrintJob class, or you can use the `print()` or `printAsBitmap()` ActionScript functions. Users can also access the Flash Player context menu and select the Print command there.

Additionally, users can print from a browser, rather than from Flash Player, by selecting a command such as File > Print from the browser window. However, printing from Flash Player directly, rather than from a browser window Print menu, offers several advantages, including the following:

- Users can print all frames or certain frames that you've labeled as printable from Flash Player. Additionally, you can set the print area of a frame.
- You can specify that content print as vector graphics (to take advantage of higher resolution) or as bitmaps (to preserve transparency and color effects).
- The ActionScript PrintJob object improves upon the `print()` and `printAsBitmap()` functions by adding the ability to print dynamically rendered pages as a single print job. The `PrintJob` object also provides the user's printer settings, which can be used to format reports specifically for the user. See "Using the ActionScript PrintJob class" on page 376.
- Flash Player versions earlier than 4.0.25 (Windows) or 4.0.20 (Macintosh) do not support printing frames directly. Flash Player 7 and later supports the PrintJob class.

This chapter contains the following sections:

Controlling printing

To control what users can print, remember the following items as you set up documents and movie clips for printing:

- Adjust the page layout in any frames that you designate as printable to match the desired printed output. Using Flash Player, you can print all shapes, symbols, bitmaps, text blocks, and text fields. Levels in a SWF file are not composited on print output.
- The Flash Player printer driver uses the HTML settings for dimension, scale, and alignment in the Publish Settings dialog box. Use these settings to control the print layout.
- The selected frames print as they appear in the movie clip symbol. You can let users print a movie clip that is not visible in a browser by setting the movie clip's `_visible` property to `false` using the Actions panel. Changing the property of a movie clip with the `setProperty` action, tweening, or any transformation tool does not affect how a movie clip prints.
- For a movie clip to be printable, it must be on the Stage or workspace and it must be given an instance name.
- All elements must be fully loaded to print. You can use the movie clip `_framesloaded` property to check whether the printable content is loaded. For more information, see `MovieClip._framesloaded` in *Flash ActionScript Language Reference*.

Supported printers

With Flash Player, you can print to PostScript and non-PostScript printers. For a list of supported Flash Player printing platforms, see the "Macromedia Flash Player Web Printing FAQ" on the Macromedia website (www.macromedia.com/software/flash/open/webprinting/faq.html).

Using the ActionScript PrintJob class

The ActionScript PrintJob class, in addition to offering improvements to print functionality available with the `print()` method, lets you also render dynamic content at runtime, prompt users with a single print dialog box, and print an unscaled document with proportions that map to the proportions of the content. This capability is especially useful for rendering and printing external dynamic content, such as database content and dynamic text.

Additionally, with properties populated by the `PrintJob.start()`function, your document can access your user's printer settings, such as page height, width, and orientation, and you can configure your document to dynamically format Flash content that is appropriate for the printer settings.

Building a print job

To build a print job, you use functions that complete the tasks in the order outlined in this section. The sections that follow the procedure provide explanations of the functions and properties associated with the PrintJob object.

Because you are spooling a print job to the user's operating system between your calls to `PrintJob.start()` and `PrintJob.send()`, and because the PrintJob functions might temporarily affect the Flash Player internal view of onscreen Flash content, you should implement print-specific activities only between your calls to `PrintJob.start()` and `PrintJob.send()`. For example, the Flash content should not interact with the user between `PrintJob.start()` and `PrintJob.send()`. Instead, you should expeditiously complete formatting of your print job, add pages to the print job, and send the print job to the printer.

To build a print job:

1. Create an instance of the print job object: `new PrintJob()`.
2. Start the print job and display the print dialog box for the operating system: `PrintJob.start()`. For more information, see "Starting a print job" on page 378.
3. Add pages to the print job (call once per page to add to the print job): `PrintJob.addPage()`. For more information, see "Adding pages to a print job" on page 378.
4. Send the print job to the printer: `PrintJob.send()`. For more information, see "Sending the print job to the printer" on page 381.
5. Delete the print job: `delete PrintJob`. For more information, see "Deleting the print job" on page 381.

The following example shows ActionScript that creates a print job for a button:

```
myButton.onRelease = function()
{
  var my_pj = new PrintJob();
  var myResult = my_pj.start();
  if(myResult){
    myResult = my_pj.addPage (0, {xMin : 0, xMax: 400, yMin: 0,
      yMax: 400});
    myResult = my_pj.addPage ("myMovieClip", {xMin : 0, xMax: 400,
      yMin: 400, yMax: 800},{printAsBitmap:true}, 1);
    myResult = my_pj.addPage (1, null,{printAsBitmap:false}, 2);
    myResult = my_pj.addPage (0);

    my_pj.send();
  }
  delete my_pj;
}
```

Only one print job can run at any given time. A second print job cannot be created until one of the following events has happened with the previous print job:

- The print job was entirely successful and `PrintJob.send()` method was called.
- The `PrintJob.start()` method returned a value of `false`.
- The `PrintJob.addPage()` method returned a value of `false`.
- The `delete PrintJob` method has been called.

Starting a print job

Calling the `PrintJob.start()` method prompts Flash Player to spool the print job to the user's operating system and also prompts the user's operating system print dialog box to appear.

If the user selects an option in the print dialog box to begin printing, the `PrintJob.start()` method returns a value of `true`. (The value is `false` if the user cancels the print job, in which case the script should call only `delete`). If successful, the `PrintJob.start()` method sets values for the `paperHeight`, `paperWidth`, `pageHeight`, `pageWidth`, and `orientation` properties.

Depending on the user's operating system, an additional dialog box might appear until spooling is complete and the function `PrintJob.send` is called: Calls to `PrintJob.addPage()` and `PrintJob.send()` should be made expeditiously. If ten seconds elapse between the `PrintJob.start()` function call and the `PrintJob.send()` function call, which sends the print job to the printer, Flash Player effectively calls `PrintJob.send()`, causing any pages that are added using `PrintJob.addPage()` to be printed and spooling to stop.

When a new print job is constructed, the `PrintJob()` properties are initialized to 0. When `PrintJob.start()` is called, after the user selects the print option in the operating system print dialog box, Flash Player retrieves the print settings from the operating system. The `PrintJob.start()` function populates the following properties:

Property	Type	Unit	Notes
`PrintJob.paperHeight`	number	points	Overall paper height.
`PrintJob.paperWidth`	number	points	Overall paper width
`PrintJob.pageHeight`	number	points	Height of actual printable area on the page; does not include any user-set margins
`PrintJob.pageWidth`	number	points	Width of actual printable area on the page; does not include any user-set margins
`PrintJob.orientation`	string	n/a	Portrait or landscape orientation

Note: A point is a print unit of measurement that is equal in size to one pixel, a screen unit of measure. For more information about unit equivalencies, see "About scaling" on page 380.

Adding pages to a print job

You add pages to your print job with the `PrintJob.addPage()` method. Although the method can include as many as four parameters, the only required parameter is `target/level`. The three optional parameters are `printArea`, `options`, and `frameNum`.

If you are not using a particular optional parameter but are using other optional parameters, use `NULL` in place of the excluded optional parameter.

With all four parameters, the function uses the following syntax:

```
MyPrintJob.addPage(target[,printArea:Object, options:Object,
  frameNum:Number]):boolean;
```

If you provide an invalid parameter, the print job uses default parameter values, which are specified in the following sections.

Each call to add a new page is unique, which lets you modify parameters without affecting previously set parameters. For example, you can specify that one page print as a bitmap image and another page print as a vector graphic. You can add as many new pages to your print job as the print job requires. One call to add a page equals one printed page.

Note: Any ActionScript that needs to be called to change a resulting printout must run before the `PrintJob.addPage()` method is called. The ActionScript can, however, run before or after a new `PrintJob()`. If a frame has a call to the `PrintJob.addPage()` method, the call itself does not guarantee that the ActionScript script on that frame will run when that frame is printed.

Specifying a target

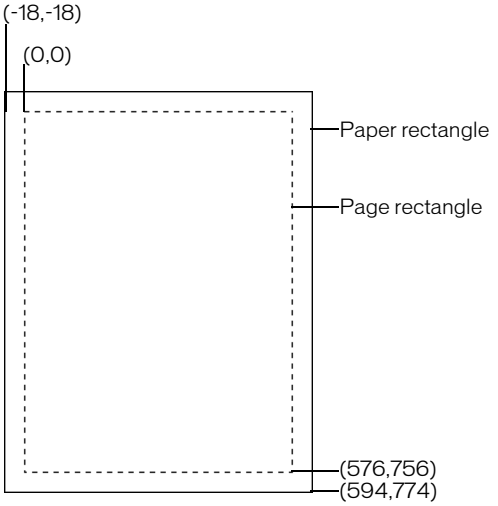
The `target` parameter can be either a number that represents a level (such as 0 for the _root document), or a string that represents the instance name of a movie clip (`"myMovieClip"`).

Specifying a print area

The `printArea` optional parameter includes the following values:

```
{xMin:Number, xMax:Number, yMin:Number, yMax:Number}
```

The `xMin`, `xMax`, `yMin`, and `yMax` values represent screen pixels relative to the target level or movie clip registration point. The print area orientation is from the upper left corner of the printable area on the page. If the print area is larger than the printable area on the page, then the print data that exceeds the right and bottom edge of the page is clipped.



If you don't specify a print area, or if you specify an invalid print area, the print area defaults to the Stage area of the root document.

About scaling

A print job using the PrintJob class prints Flash content, by default, without scaling it. For example, an object that is 144 pixels wide on screen will print as 144 points, or 2 inches wide (One point equals one pixel. In the authoring tool, 72 pixels equals one inch; on paper, 72 points equals one inch.)

To understand how Flash screen content maps to the printed page, it helps to understand screen and print units of measure. Pixels are a screen measurement and points are a print measurement. Both pixels and points equal 1/72 of an inch. A *twip* is 1/20 of a point and pixel.

The following list further illustrates the relationship between units of measure.

- 1 pixel = 20 twips
- 1 point = 20 twips
- 72 pixels = 1 inch
- 72 points = 1 inch
- 567 twips = 1 cm
- 1440 twips = 1 inch

To scale a movie clip before printing, set its `MovieClip.xscale` and `MovieClip.yscale` properties before calling this method, and set them back to their original values afterward. If you scale a movie clip and also pass a value for the *`printArea`* property, the pixel values passed to *`printArea`* reflect the original size of the movie clip. That is, if you set a movie clip's scale to 50% and specify a print area of 500 x 500 pixels, the content that prints is identical to the content that would print if you didn't scale the movie clip; however, it prints at half the size. For more information, see `PrintJob.addPage()` in *Flash ActionScript Language Reference*.

Specifying printing as a vector image or bitmap graphic

The `options` parameter lets you specify whether to print as a vector graphic or bitmap image. When using this optional parameter, use the following syntax:

```
{printAsBitmap:boolean}
```

The default value is `false`, which represents a request for vector printing. Remember the following suggestions when determining which value to use:

- If the content that you're printing includes a bitmap image, you should specify that the print job print as a bitmap to include any transparency and color effects.
- Conversely, if the content does not include bitmap images, you should specify that the print job print as vector graphics to take advantage of the higher image quality.

Specifying a frame to print

The `frameNum` parameter lets you specify a frame to print. If you do not specify a frame number parameter, the current frame of the target or level specified as the first parameter when adding a page prints by default.

Sending the print job to the printer

To send the print job to the printer after using the `addPage()` calls, use the `PrintJob.send()` method, which causes Flash Player to stop spooling the print job so that the printer starts printing.

Deleting the print job

After sending the print job to a printer, use the ActionScript function `delete PrintJob()` to delete the PrintJob object, which frees memory. For more information, see `delete` in *Flash ActionScript Language Reference*.

Printing frames independent of the PrintJob class

The PrintJob class, available for Flash Player 7 and later, offers many advantages over the `print()` and `printAsBitmap()` methods for printing. However, to print targeting Flash Player 6 and earlier versions, back to Flash Player 4.0.25 (Windows) and 4.0.20 (Macintosh), you can use `print()` and `printAsBitmap()` functions and frame labels—classic functionality that remains part of the authoring tool and does not use the PrintJob class.

To set up printing from Flash Player independent of the PrintJob class, you can specify frames to print and set their print area.

For more information on use of the PrintJob class, see "Using the ActionScript PrintJob class" on page 376.

Designating printable frames (when not using the PrintJob object)

All frames in the specified Timeline print by default. You might want to limit the number of frames that print—for example, if you have a lengthy animation of dozens of frames. You can designate specific frames in a SWF file as printable in order to print only those frames; unspecified frames won't print.

To specify frames as printable, you label the frames.

To designate printable frames:

1. Open or make active the SWF file that you want to publish.
2. Select the desired frame in the Timeline that you want to make printable and add a keyframe.
3. In the Property inspector (Window > Properties), enter **#p** in the Label text box to specify the frame as printable.
4. Repeat steps 2 and 3 for each frame you want to designate as printable.

Note: If you have multiple #p labels in your document, you might receive an Output window message when you test or publish your SWF file that indicates that the document contains duplicate frame labels. You can ignore the message if the duplicate labels are all #p labels.

To control what users can print, remember the following items as you set up documents and movie clips for printing:

- Adjust the page layout in any frames that you designate as printable to match the desired printed output. Using Flash Player, you can print all shapes, symbols, bitmaps, text blocks, and text fields. Levels in a SWF file are not composited on print output.
- The Flash Player printer driver uses the HTML settings for dimension, scale, and alignment in the Publish Settings dialog box. Use these settings to control the print layout.
- The selected frames print as they appear in the movie clip symbol. You can let users print a movie clip that is not visible in a browser by setting the movie clip's `_visible` property to `false` using the Actions panel. Changing the property of a movie clip with the Set Property action, tweening, or any transformation tool does not affect how a movie clip prints.
- For a movie clip to be printable, it must be on the Stage or workspace and it must be given an instance name.
- All elements must be fully loaded to print. You can use the movie clip `_framesloaded` property to check whether the printable content is loaded. For more information, see `MovieClip._framesloaded` in *Flash ActionScript Language Reference*.

Specifying a print area (when not using the PrintJob object)

By default, when frames are printed, the document file's Stage determines the print area. Any object that extends off the Stage is clipped and does not print. Loaded movies use their own Stage size for the print area, not the main movie's Stage size.

As an alternative to using a document's Stage size, you can set the following print areas:

- For either the Flash Player context menu or the `print()` function, you can designate the SWF content bounding box as the print area for all frames by selecting an object in one frame as the bounding box. This option is useful, for example, if you want to print a full-page data sheet from a web banner.
- With the `print()` function, you can use the composite bounding box of all printable frames in a Timeline as the print area—for example, to print multiple frames that share a registration point. To use the composite bounding box, you use the `bMax` parameter, as shown in the following example:

  ```
  print ("myMovie", "bmax")
  ```

- With the `print()` function, you can change the print area for each frame, scaling objects to fit the print area—for example, to have objects of different sizes in each frame fill the printed page. To change the bounding box per frame, use the Frame parameter in the Print action parameters, as shown in the following example:

  ```
  print ("myMovie", "bframe")
  ```

- With the `print()` function, you can designate the bounding box of a specific frame in a document as the print area for all printable frames in the document, as shown in the following example:

  ```
  print ("myMovie", "bmovie")
  ```

You use the label #b to designate a frame to be used to designate the print area. The label #b must be on the same layer as a frame labeled #p.

For more information about `print()` function parameters, see `print()` in *Flash ActionScript Language Reference*.

To specify a print area when printing frames:

1. Open the Flash document (FLA) containing the frames you will set to print.
2. Select a frame that you have not specified to print with a #p frame label. Select a frame that is on the same layer as one labeled #p.

   To organize your work, you can select the next frame after one labeled #p.
3. Create a shape on the Stage the size of the desired print area.

   You can also select a frame with any object of the appropriate print area size to use that frame's bounding box.
4. Select the frame in the Timeline that contains the shape you want to use for the bounding box.
5. If the Property inspector does not appear, select Window > Properties.
6. In the Property inspector, enter **#b** for Label to specify the selected shape as the bounding box for the print area.

   You can enter only one #b label per Timeline. This option is the same as selecting the Movie bounding box option with the Print action.

Using the print() function (when not using the PrintJob object)

The basic syntax for the `print()` function, which you can associate with a button or other trigger in your document to activate printing, is shown as follows:

```
print (target, "Bounding box");
```

The target parameter specifies the location of the frames that print, and the bounding box parameter specifies the print area.

You can add a `print()` function to a button or other element in your document to let users print Flash content. You assign the `print()` function to a button, frame, or movie clip. If you assign a `print()` function to a frame, the action executes when the playhead reaches the designated frame.

The `print()` function lets you print frames in other movie clips in addition to the main Timeline. Each `print()` function sets only one Timeline for printing, but the action lets you specify any number of frames within the Timeline to print. If you attach more than one `print()` function to a single button or frame, the Print dialog box appears for each action executed. For more information about the `print()` function, see `print()` in *Flash ActionScript Language Reference*.

Changing the printed background color

With Flash Player, you can print the background color set in the Document Properties dialog box. You can change the background color for only the frames to be printed by placing a colored object on the lowest layer of the Timeline being printed.

To change the printed background color:

1. Place a filled shape that covers the Stage on the lowest layer of the Timeline that will print.
2. Select the shape and select Modify > Document. Select a color for the printing background.

   This changes the entire document's background color, including that of movie clips and loaded movies.
3. Do one of the following:
   - To print that color as the document's background, make sure that the frame in which you placed the shape is designated to print. For instructions, see "Specifying a frame to print" on page 380 or "Using the print() function (when not using the PrintJob object)" on page 383.
   - To maintain a different background color for nonprinting frames, repeat steps 2 and 3. Then place the shape on the lowest layer of the Timeline, in all the frames that are not designated to print.

Using frame labels to disable printing

If you don't want any of the frames in the main Timeline to print, you can label a frame as !#p to make the entire SWF file nonprintable. Labeling a frame as !#p dims the Print command in the Flash Player context menu. You can also remove the Flash Player context menu.

If you disable printing from Flash Player, the user can still print frames using the browser Print command. Because this command is a browser feature, you cannot control or disable it using Flash.

To disable printing in the Flash Player context menu by dimming the Print command:

1. Open or make active the Flash document (FLA file) that you want to publish.
2. Select the first keyframe in the main Timeline.
3. Select Window > Properties to view the Property inspector.
4. In the Property inspector, for Label enter **!#p** to specify the frame as nonprinting.

   You need to specify only one !#p label to dim the Print command in the context menu.

   Note: You can also select a blank frame (rather than a keyframe) and label it **#p**.

To disable printing by removing the Flash Player context menu:

1. Open or make active the Flash document (FLA file) that you want to publish.
2. Select File > Publish Settings.
3. Select the HTML tab and deselect Display Menu.
4. Click OK.

For more information on publishing options, see "Publishing Flash documents" on page 311.

Printing from the Flash Player context menu

You can use the Print command in the Flash Player context menu to print frames from any Flash SWF file.

The context menu's Print command cannot print transparency or color effects and cannot print frames from other movie clips; for more sophisticated printing capabilities, use the PrintJob object or the `print()` function. See "Using the ActionScript PrintJob class" on page 376 and "Using the print() function (when not using the PrintJob object)" on page 383.

To print frames using the Flash Player context menu Print command:

1. Open the document with frames you want to print.

   The command prints the frames labeled #p using the Stage for the print area or the specified bounding box.

   If you haven't designated specific frames to print, all frames in the document main Timeline print.
2. Select File > Publish Preview > Default or press F12 to view your Flash content in a browser.
3. Right-click (Windows) or Control-click (Macintosh) in the Flash content in the browser window to display the Flash Player context menu.
4. Select Print from the Flash Player context menu to display the Print dialog box.
5. In Windows, select the print range to select which frames to print:
   - Select All to print all frames if no frames are labeled.
   - Select Pages and enter a range to print the labeled frames in that range.
   - Select Selection to print the current frame.
6. On the Macintosh, in the Print dialog box, select the pages to print:
   - Select All to print the current frame if no frames are labeled or to print all labeled frames.
   - Select From and enter a range to print the labeled frames in that range.
7. Select other print options, according to your printer's properties.
8. Click OK (Windows) or Print (Macintosh).

Note: Printing from the context menu does not interact with calls to the `PrintJob` object.

Publishing a document with printable frames

You can publish a Flash document with printable frames to the web using the Publish command to generate the necessary Flash HTML templates. For more information, see "Publishing Flash documents" on page 311.

Users must have Flash Player 4.0.25 (Windows) or 4.0.20 (Macintosh) or later to take advantage of any print functionality you have added and to print the designated frames in Flash. You can set up a detection scheme to check for the proper Flash Player version.

Note: When you use the PrintJob class, users must have Flash Player 7 or later.

CHAPTER 19
Creating E-learning Content

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 learning interactions help you create interactive online (e-learning) courses that run in Flash. Using the Flash learning interactions has many benefits:

- Anyone with a Flash-enabled web browser can use the instructional content you create.
- You can customize the interface to meet your needs. Because you are using Flash, you can create high-quality interfaces that load quickly and look the same on different platforms.
- You can easily add interactions to your online course with the Flash Learning Interaction components, which provide a simple interface for entering data without writing code.
- Each individual Flash learning interaction can send tracking information to a server-side learning management system (LMS) that complies with the Aviation Industry CBT Committee (AICC) protocol or Shareable Content Object Reference Model (SCORM) standards.
- Additionally, the quiz templates track cumulative results from a sequence of interactions and can pass them along to the LMS using an enhanced data tracking functionality that conforms to either AICC or SCORM standards.

This chapter contains the following sections:

Getting started with Flash learning interactions

Your e-learning courseware runs on any computer with Macromedia Flash Player 6 or later and a Flash-enabled web browser.

To track user data from the Flash learning interactions, you must have a web server-side LMS, such as an AICC- or SCORM-compatible system. In addition, users must have Internet Explorer 4.0 or Netscape Navigator 4.0 or later (Windows), or Netscape 4.5 or later (Macintosh). Tracking to an LMS with learning interactions does not work with Internet Explorer on the Macintosh.

About Flash learning interactions

An interaction is a part of a Flash application in which the user interacts with the application to provide a response. A typical response might be answering a question, selecting from the answers True or False, or clicking an area of the screen. You can use the six learning interactions included with Flash to build interactive courseware:

True or False In this type of interaction, the user responds to a question with the answers True or False.

Multiple Choice The user responds to a multiple-choice question.

Fill in the Blank The user types a response that is checked against matching phrases.

Drag and Drop The user responds to a question by dragging one or more onscreen objects to a target.

Hot Spot The user responds by clicking a region (or regions) on the screen.

Hot Object The user responds by clicking an object (or objects) on the screen.

Each learning interaction has unique parameters that determine how the interaction appears to the user. The interactions are Flash components, which makes them easy to implement and configure in a Flash document. For additional information about Flash components, see Chapter 5, “Customizing Components” in *Using Components*.

Including a Flash learning interaction in a document

You can use either quiz templates or stand-alone interactions in your Flash documents:

- The quiz templates are designed for scenarios in which interaction-based quizzes are required or tracking is necessary. The quiz learning interactions are graphically designed to fit into the quiz format. The quiz templates contain a mechanism that counts a cumulative score and starts and stops the necessary tracking in both AICC- and SCORM-compliant APIs.
- The stand-alone interactions are designed for scenarios that require a single interaction, or a series of interactions that need to fit into a specific layout within a Flash document. These are available from the common library and are graphically designed for stand-alone use. You can track the results for each individual stand-alone interaction and submit them to an AICC-compatible LMS. See "Adding learning interactions to a quiz template" on page 395.

To initialize SCORM tracking, you must use a quiz template.

Using the quiz templates

Each of the three quiz templates that come with Flash has a different graphical look and feel, but they are otherwise identical. Each template contains the following elements:

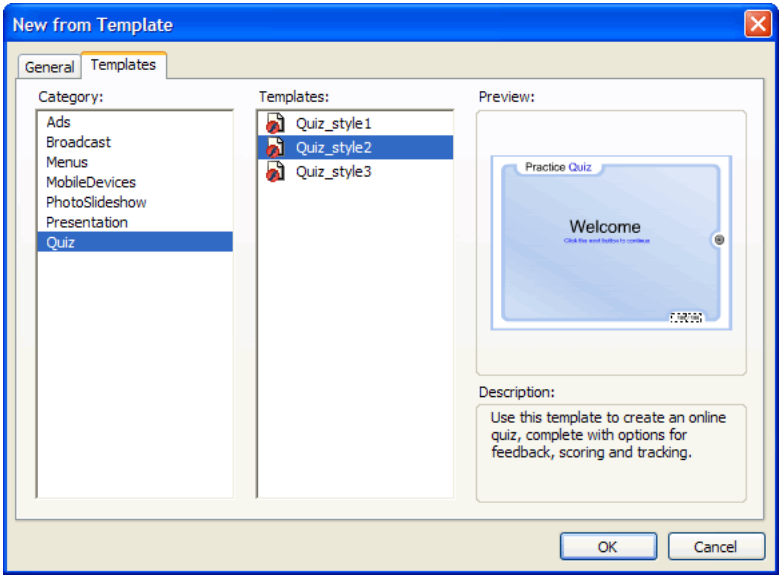
- A Welcome page
- One of each of the six learning interaction types
- A Results page
- Navigation elements
- ActionScript to gather AICC and SCORM tracking information

The quiz templates provide built-in navigation to move among interactions. They also include ActionScript that can pass tracking information to a web server.

The quiz templates are fully functional. After creating a new document from a quiz template, you can immediately test the document, before modification, to see how the quiz functions. Included with a quiz are each of the six learning interaction types that are stored in movie clips in the library. These movie clips are simply containers for the collection of elements that comprise each interaction. You break apart the movie clips to edit the pieces.

To create a quiz:

1. Create a new file by selecting File > New.
2. In the New from Template window, select the Templates tab.
3. In the Category column, select Quiz; then in the Templates column, select one of the quiz styles.

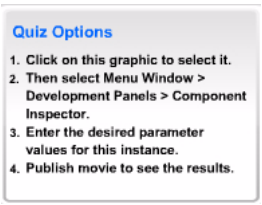


Setting the quiz parameters

After you create a new file and select one of the quiz templates, the next step is to set the quiz parameters. These parameters control how the entire quiz is presented to users—for example, whether the questions are presented in a random or sequential order, the number of questions to display, and whether the Results page appears.

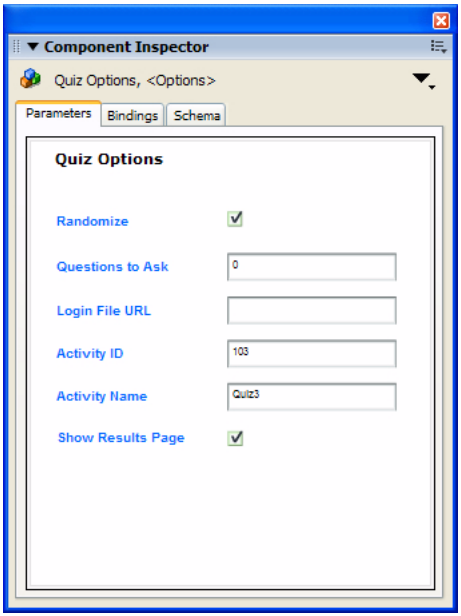
To set quiz parameters:

1. Select the Quiz Options component with instructions to the left of the Stage in the quiz template. The component lets you set the parameters for the quiz.



Note: These instructions do not appear in the SWF file.

2. Do one of the following to open the Component Inspector:
   - Select Window > Development Panels > Component Inspector.
   - In the Property inspector, click Launch Component Inspector.



Note: If the text in the Component inspector is too small to be legible, drag a corner of the panel to enlarge it. You may need to undock the panel to enlarge it.

3. Select Randomize if you want the quiz questions to be presented in a random order and not necessarily in the order in which they appear in the Timeline.
4. In the Questions to Ask text box, specify the number of questions to ask for one presentation of the quiz. If you set this number to 0, the quiz uses all the questions you add to the document. If you enter a number larger than the number of questions in the quiz, the quiz displays only the number of questions that are in the quiz and does not duplicate any of them.

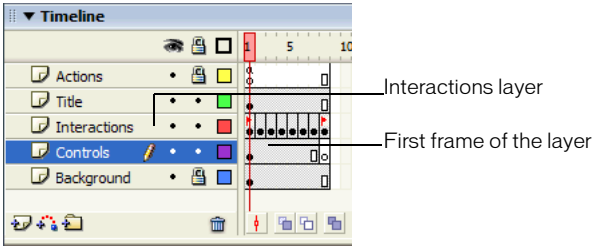
   For example, if you have 10 interactions in your quiz, you can specify that a lesser number, such as 5 interactions, appear to the user. This feature is especially helpful when used with the Randomize feature to create quizzes with unexpected questions in an unexpected order.
5. Enter the URL to redirect the user.

   When an AICC-compliant LMS starts a quiz, it includes parameters that the HTML code looks for when it executes the `embed` tag for the Flash application and the course loads properly. If no parameters are specified, the user is redirected to the URL specified in the Login File URL field. If this field is blank or the Flash file was published with the SCORM template, the redirect does not occur.

6. In the Activity ID and Activity Name text boxes, enter the activity ID and activity name of your LMS, if you are using one. If you are not using an LMS, you can either accept or delete the default entries.
7. Select Show Results Page if you want to present quiz results to users after they have completed the quiz.

Modifying learning interactions in a quiz

Each question in the quiz is considered an interaction. When you use a quiz template, you place interactions sequentially between the first and last frame of the Interactions layer on the root Timeline. You may add or remove frames and keyframes as needed, as long as the interactions remain sequential and the first and last frames are reserved for the Welcome page and Results page. The number of frames between the Welcome and Results page keyframes are used to calculate the score.



For example, the following frames would comprise a 10-question quiz:

- Frame 1 = Welcome page keyframe
- Frames 2–11 = interactions keyframes
- Frame 12 = Results page keyframe

These 12 keyframes are on the Interactions layer.

To modify learning interactions in a quiz template:

1. Select the first frame in the Interactions layer and make any modifications you want to the text of the Welcome page. Make sure you include text to indicate that the user must click the Next button to continue. Do not add an interaction to this page.
2. Select each of the learning interactions in the next six frames and do one of the following:
   - If you want to use the interaction, follow the instructions in "Configuring a Learning Interaction component" on page 393.
   - If you do not want to use the interaction, follow the instructions in "Removing a learning interaction from the Timeline" on page 397.
3. Select the last frame in the Interactions layer and make any modifications you want to the text of the Results page. Make sure to leave the supplied dynamic text field names intact, though, or the results will not appear. Do not delete or place interactions in this frame. If the Results Page quiz parameter is turned off for the quiz, this frame is not called, but it is still reserved.

Configuring a Learning Interaction component

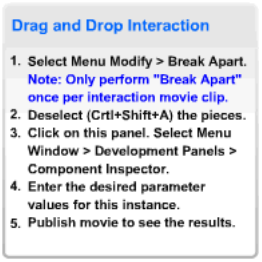
Included with each quiz template is one of each of the six learning interaction types, stored in movie clips in the library. These movie clips are simply containers for the collection of elements that make up each interaction. When you add an interaction (movie clip) to the Stage, you must break it apart to edit the individual objects.

To configure a Learning Interaction component:

1. With the entire learning interaction selected, select Modify > Break Apart. This breaks the interaction into individual objects that can be modified.

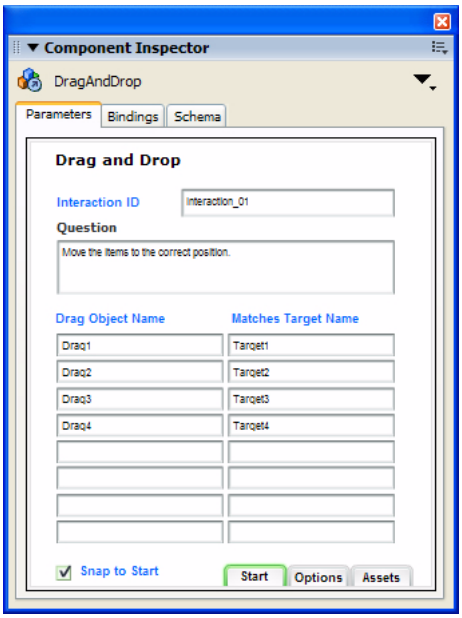
   Note: Make sure that you break apart the interaction only once. See "Testing to see if a movie clip is broken apart" on page 398.

2. Deselect all the items on the Stage (Control+Shift+A).
3. Select the Learning Interaction component.



Note: Do not delete these instructions from the document; they contain necessary ActionScript code and do not appear in the SWF file.

4. In the Property inspector, click Launch Component Inspector.



5. If the Flash application will send tracking information to a server-side LMS, specify a name for the interaction in the Interaction ID text box. You should uniquely name each interaction in the quiz as specified by your LMS. Each interaction in the quiz templates is uniquely named. However, if you add interactions from the library or you are not using the quiz template, make sure to uniquely name each interaction in your file.
6. In the Question text box, type the text you want the user to see. This text can be a question and/or instructions for the user.
7. Configure the learning interaction. For more information, see the following sections:
   - "Configuring a Drag and Drop interaction" on page 400
   - "Configuring a Fill in the Blank interaction" on page 402
   - "Configuring a Hot Object interaction" on page 403
   - "Configuring a Hot Spot interaction" on page 404
   - "Configuring a Multiple Choice interaction" on page 406
   - "Configuring a True or False interaction" on page 407
8. At the bottom of the Component Inspector, Click Options and enter feedback and Knowledge Track parameters for the learning interaction. See "Adding, naming, and registering assets" on page 407, "Setting Knowledge Track options for a learning interaction" on page 412, and "Setting navigation options for a learning interaction" on page 413.

   Note: Documents created using a quiz template have the Knowledge Track option turned on and the Navigation option turned off (the default settings) for each learning interaction, because the quiz template has its own navigation controls.

9. (Optional) Click the Assets button, and change the assets for the learning interaction. See "Adding, naming, and registering assets" on page 407.

Adding learning interactions to a quiz template

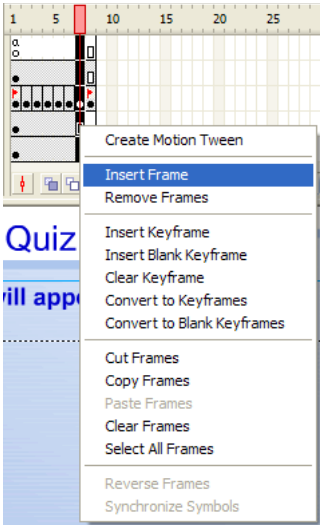
When you use a quiz template, you add learning interactions to the Interactions layer.

To add an interaction to the Timeline when using a quiz template:

1. In the first layer of the Timeline, select the frame that precedes the frame number in which you want to add the interaction.

   For example, if you want to add an interaction to Frame 8, select Frame 7.

2. Shift-click the same frame number on the other layers to also select those frames.
3. Right-click (Windows) or Control-click (Macintosh) a selected frame and select Insert Frames to extend the Timeline evenly across all layers.



4. On the Interactions layer, select the frame you added and select Insert > Timeline > Blank Keyframe.
5. To add an interaction, do one of the following:

   - To copy and paste an interaction that already exists on the Timeline, right-click (Windows) or Control-click (Macintosh) the keyframe with the interaction and select Copy Frames. Paste the frame in the blank keyframe that you inserted in step 4. In this copy of the interaction, modify objects on the Stage or the settings in the Component Inspector, as desired.
   - To use an interaction from the library, drag the desired interaction movie clip type from the Learning Interactions library (Window > Other Panels > Common Libraries > Learning Interactions) to the blank keyframe. Break the interaction apart (select the interaction and select Modify > Break Apart), and edit the assets and parameters.

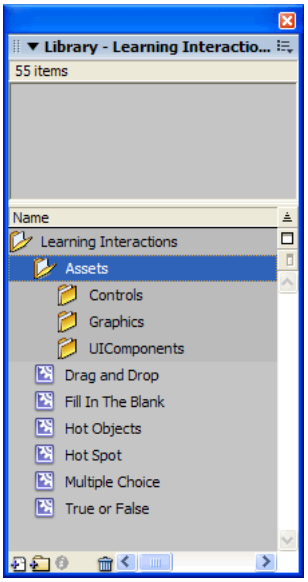
Adding learning interactions to a document that doesn't use a quiz template

If you are adding learning interactions to a Flash document that does not use a quiz template, you can place stand-alone learning interactions on the Timeline in a single frame, sequential frames (for example, 10 questions in 10 sequential frames), or labeled frames.

To add a stand-alone learning interaction to the Timeline when not using a quiz template:

1. If you are adding interactions to a document that does not use the quiz template, select the appropriate layer, and then select Insert > Timeline > Blank Keyframe.
2. Select Window > Other Panels > Common Libraries > Learning Interactions.

   The Learning Interactions library appears.

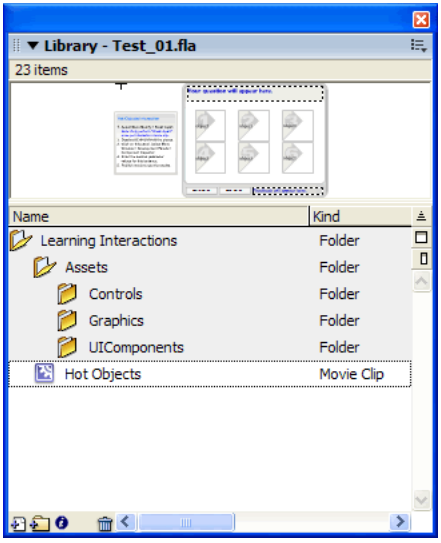


   The library includes six types of learning interaction movie clips: Drag and Drop, Fill in the Blank, Hot Object, Hot Spot, Multiple Choice, and True or False. In addition, there are folders called Assets, Graphics, and UIComponents. These are used for customizing learning interactions. See "Changing buttons, check boxes, and radio buttons" on page 399.
3. Select the new keyframe you created, and then drag one of the Learning Interaction movie clips from the Library panel to the Stage.
4. Reposition the interaction by dragging it to where you want it to appear on the Stage.
5. Configure the learning interaction. See "Configuring a Learning Interaction component" on page 393.

   Note: Watch the frame count across layers as you add and remove keyframes. Make sure that all layers end at the same frame number along the Timeline so that the frame count is the same in all layers.

About managing library assets for learning interactions

When you drag a learning interaction from the Learning Interactions common library to the Stage, the symbols that comprise the learning interaction are copied from the common library to the library of the Flash document you are creating. For example, if you copy a Hot Object learning interaction from the Learning Interactions common library to your document, the symbols in the following illustration become part of the document library.



If you're using a quiz template, the learning interaction symbols are already included in your document library.

To manage library assets, it is a good idea to create folders for each graphical interaction and place the folders within the Assets folder. You can then keep the movie clips associated with the interaction within the new folder.

Removing a learning interaction from the Timeline

When you remove a learning interaction from the Timeline, it is important to maintain the sequence of learning interactions. If you remove a frame from the Interactions layer, you also need to remove it from all other layers.

To remove an interaction from the Timeline:

1. On the Interactions layer, select the keyframe containing the interaction to be deleted. Shift-select the same frame number on other layers if you want to also delete those frames.
2. To delete frames across all layers, do one of the following:
   - Right-click (Windows) or Control-click (Macintosh) the keyframe and select Remove Frames.
   - Select Edit > Timeline > Remove Frames.

Note: Watch the frame count across layers as you add and remove keyframes. Make sure that all layers end at the same frame number along the Timeline so that the frame count is the same in all layers.

Testing to see if a movie clip is broken apart

It is a good idea to check whether a learning interaction is broken apart or still grouped within the movie clip container.

To verify whether a learning interaction is broken apart:

- Select a text field or any other single element of the learning interaction on the Stage.

  If a grouped object is selected, the interaction is not broken apart.

  If you can select a single text field or another element, then the interaction has been broken apart and you can proceed with editing.

Changing the appearance of a learning interaction

After you have added a learning interaction to the Stage and broken it apart, you can place and size most assets the same as you would in any other Flash document. For example, you can extend text fields so they can accommodate more lines of text, and you can adjust the font, size, color, and other text properties. However, making changes to certain Flash components, such as buttons, check boxes, and radio buttons within learning interactions, requires less common processes. See "Changing buttons, check boxes, and radio buttons" on page 399.

Changing the images in a graphical learning interaction

For Drag and Drop, Hot Spot, and Hot Object learning interactions, you change the appearance of the graphic *distractors* (the selectable choices) in the interaction to suit the purposes of your course.

To change the images in a graphical learning interaction:

1. If it has not been broken apart, select the learning interaction movie clip and the select Modify > Break Apart.
2. Select the placeholder graphical objects, such as the four placeholder Drag objects and four placeholder Target objects, and delete them.
3. To add your own custom Drag object(s), create or import a graphic and convert it to a movie clip symbol (Modify > Convert to Symbol).
4. Place an instance of the symbol in the desired location on the Stage. In the Property inspector, type the name of the movie clip instance, such as DragA, in the Instance Name text box.
5. In the Component inspector for the interaction, enter the same instance name (such as DragA) of the movie clip in the appropriate Name text box. The Component inspector should include only the unique instance names of the movie clips that you're using for the current interaction.
6. Repeat steps 3–5 for additional graphical objects within the interaction.

Note: The graphics for navigation buttons and for True or False and Multiple Choice interactions are created using Flash UI components. Only intermediate and advanced users should change these graphics. For more information, see Chapter 5, "Customizing Components" in *Using Components*.You can also resize and slightly modify the appearance of these graphics. See "Changing buttons, check boxes, and radio buttons".

Changing buttons, check boxes, and radio buttons

The learning interactions use the Flash user interface (UI) Button, CheckBox, RadioButton and TextInput components. You must use these UI components within the learning interaction movie clips. The learning interaction scripts use the internal features of the UI components to function properly.

The quiz templates already contain all the necessary UI components for each interaction. To use UI components in Flash MX or later documents, you must publish the SWF file using ActionScript 2.0.

Sizing

The Button components used for the Control button and Reset button can be scaled to fit your needs, as can the CheckBox, RadioButton, and TextInput components.

To set the width and height of the Button, CheckBox, and RadioButton components:

- Select the component and change its settings in the Property inspector.

UI component graphics

There is a defined process for changing the skin of a component. For more information, see "Editing component skins in a document" in *Using Components*.

UI component text

You can use the `GlobalStyleSheet` object to change the text characteristics of a UI component. For detailed information, see "Setting control button labels for a learning interaction" on page 414. See also Chapter 5, "Customizing Components" in *Using Components*.

About using components within a learning interaction

To use Flash UI components with a learning interaction, you simply add the UI components to the interaction assets and name their instances. You then need to register the instance names with the component associated with that interaction. Each learning interaction already contains the appropriate UI components as named instances. See "Adding, naming, and registering assets" on page 407.

For complete documentation on the UI components, see *Using Components*.

Note: UI components have a Component inspector associated with them. The learning interaction scripts override the UI Component inspector at runtime. There is no need to fill out individual parameters for each Button, CheckBox, RadioButton or TextInput component.

Testing a quiz

It is important to test a quiz frequently as you add and remove interactions.

To test a quiz:

1. Select Control > Test Movie.

   The quiz appears in the Flash Player window.

2. Answer the questions as they appear.

3. When you complete the quiz, close it in the Flash Player window to return to the workspace in which you edit the document.

Configuring learning interactions

For each of the six interactions, you must enter specific parameters for the quiz to function properly. A Drag and Drop interaction requires that you specify the Target object and the Drag object. Each Target object and Drag object is referred to as a *distractor*. A distractor is simply one of a series of selectable choices. This term is used for the choices in each of the learning interactions. For example, with a Multiple Choice learning interaction, you enter the multiple-choice distractors.

Configuring a Drag and Drop interaction

You can use as many as eight Drag objects and eight Target objects in each Drag and Drop interaction. Each Drag object can snap to any target named in the Drag and Drop component for evaluation. Drag objects can also share targets; for example, both Drag 1 and Drag 2 can match Target 8. You can also specify a target without matching a Drag object to it, which lets you add incorrect target distractors for evaluation.

Each Target object and Drag object is referred to as a *distractor*. A distractor is simply one of a series of selectable choices. This term is used for the choices in each of the learning interactions.

To configure a Drag and Drop interaction:

1. If you are not using a quiz template, place the learning interaction on the Stage. If you are using a quiz template, select the frame on the Interactions layer that contains the Drag and Drop interaction (Frame 2, if you have not added or removed keyframes).

2. Break the movie clip apart (Modify > Break Apart), display the Component inspector, and then type the interaction ID and the question. See "Configuring a Learning Interaction component" on page 393.

3. In the Drag Object Name column, list the instance names for the Drag objects on the Stage.

   Each Drag object must have a unique name. If you add a new Drag object on the Stage, make sure to enter its name here.

4. In the Matches Target Name column, list the matching target instance name for that Drag object.

   Each target must have a unique name. If you add a new target on the Stage, make sure to enter its name here.

   If you enter a Drag instance name in the Drag Object Name column, you need to enter a corresponding Target instance name in the Matches Target Name column. However, you can enter a Target instance name in the Matches Target Name column without a matching Drag instance name. This adds a target that can be snapped to but is not evaluated as a correct match.

5. Select Snap to Start to make the Drag objects snap back to their original position if they do not snap to a registered target.

6. Select each instance of the Drag object or Target object on the Stage. Use the Property inspector to give each instance the same instance name that you specified in the Component inspector.

Adding and removing Drag objects and Target objects

You can change the default number of four objects and four targets by adding more objects and targets or by deleting existing objects and targets. You can include from one to eight Drag objects and one to eight Target objects in a Drag and Drop learning interaction.

To add a Drag object or Target object:

1. Create a movie clip symbol containing the graphics for the object. For example, if you have an interaction that has six types of fruit, and you want to add a seventh choice, create a graphic of the seventh fruit and place it in the library.

2. Select the Drag and Drop learning interaction on the Timeline, and then drag the symbol from the Library panel to the Stage.

3. In the Property inspector, name the instance. See "Adding, naming, and registering assets" on page 407.

4. Add the instance name to the Component inspector for the Drag and Drop object. See "Naming and registering graphic distractors" on page 409.

   The component does the rest of the work automatically at runtime.

To remove a Drag and Drop object:

1. Select the Drag and Drop instance that you want to remove, and delete it from the Stage.

2. Select the Drag and Drop component (to the left of the Stage in the quiz template), and then display the Component inspector by opening it from the Property inspector, if necessary.

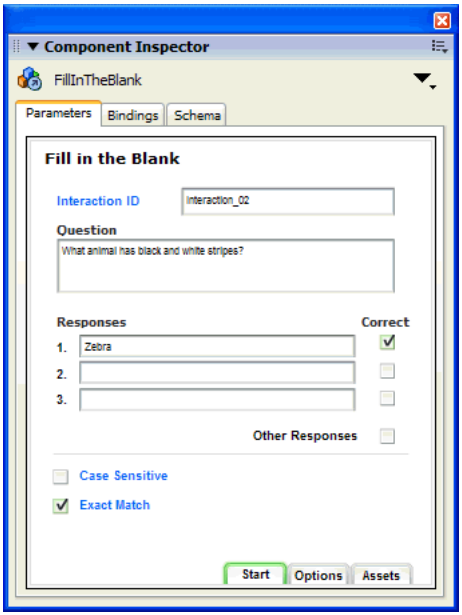
3. Remove the deleted object's instance name from the appropriate column in the Component Inspector.

Configuring a Fill in the Blank interaction

The Fill in the Blank interaction uses a question text field, a user entry text field, a control button, and a feedback text field.

To set up a Fill in the Blank interaction:

1. If you are not using a quiz template, place the learning interaction on the Stage. If you are using a quiz template, select the frame on the Interactions layer that contains the Fill in the Blank interaction (Frame 3, if you have not added or removed keyframes).
2. Break the movie clip apart (Modify > Break Apart), display the Component Inspector, and then type the interaction ID and the question. See "Configuring a Learning Interaction component" on page 393.
3. In the Component Inspector, do one of the following to enter one to three possible correct answers:



   - Type the text for the responses that the user can enter that are considered correct responses. Select the Correct option to the right of the correct responses.
   - To set up the interaction to accept all responses except those you type, enter the invalid responses in the list and deselect the Correct option to the right of them. Then select the Other Responses option, to indicate that all other responses are correct.
4. Specify whether the matching responses are valid only if they match the case of the text you entered (by selecting Case Sensitive) or if they are valid regardless of the capitalization the user enters (by deselecting Case Sensitive).

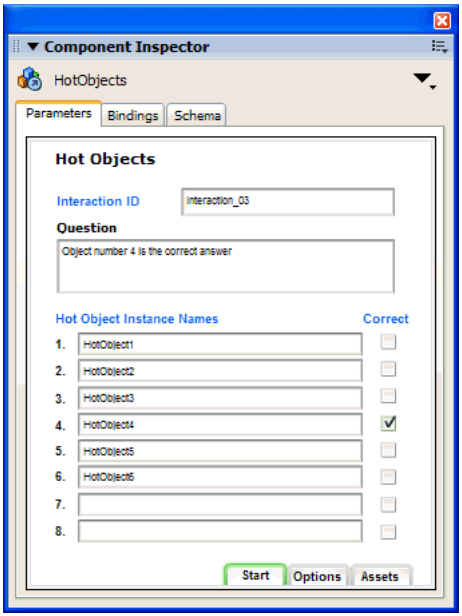
5. Specify whether the matching response must be an exact match. If you select Exact Match, a correct response matches only if the user enters the text exactly as it appears in your response. With Exact Match deselected, an answer is considered correct if it contains the correct word. For example, if the answer is zebra and the user enters **striped zebra**, the answer is considered correct. This feature does not work if the correct answer is more than one word.

Configuring a Hot Object interaction

The Hot Object interaction accepts one to eight hot objects. The default sample uses six hot objects.

To configure a Hot Object interaction:

1. If you are not using a quiz template, place the learning interaction on the Stage. If you are using a quiz template, select the frame on the Interactions layer that contains the Hot Object interaction (Frame 5, if you have not added or removed keyframes).
2. Break the movie clip apart (Modify > Break Apart), display the Component Inspector, and then type the interaction ID and the question. See "Configuring a Learning Interaction component" on page 393.



3. For each object, select or deselect the Correct option to specify whether the object is considered a correct or incorrect response when the user clicks it. You can have multiple correct selections.
4. Select each instance of the Hot Object interaction on the Stage (you can delete the placeholder instances and place your own movie clip instances on the Stage). Use the Property inspector to give each instance the same instance name that you specified in the Component Inspector.

Adding and removing hot object distractors

You can change the default number of six distractors (choices) by adding more distractors or deleting existing distrators. You can include from one to eight hot object distractors in a Hot Object learning interaction.

To add a hot object distractor:

1. Create a movie clip symbol containing the graphics for the hot object distractor. For example, if you have an interaction that has six types of fruit, and you want to add a seventh choice, create a graphic of the seventh fruit and place it in the library.
2. Select the Hot Object component on the Stage, and then drag the symbol from the Library panel to the Stage.
3. In the Property inspector, name the instance. See "Naming and registering graphic distractors" on page 409.
4. Add the instance name to the Component inspector for the hot object.

   The component does the rest of the work automatically at runtime.

To remove a hot object distractor:

1. Select the Hot Object movie clip instance that you want to remove, and delete it from the Stage.
2. Select the Hot Object component (to the left of the Stage in the quiz template), and then display the Component inspector by opening it from the Property inspector, if necessary.
3. Remove the deleted object's instance name from the list in the Component Inspector.

Configuring a Hot Spot interaction

The Hot Spot learning interaction sets up an interaction in which the user responds by clicking an object (or objects) onscreen.

An example of a Hot Spot interaction created with the quiz template

To configure a Hot Spot interaction:

1. If you are not using a quiz template, place the learning interaction on the Stage. If you are using a quiz template, select the frame on the Interactions layer that contains the Hot Spot interaction (Frame 5, if you have not added or removed keyframes).
2. Break the movie clip apart (Modify > Break Apart), display the Component Inspector, and then type the interaction ID and the question. See "Configuring a Learning Interaction component" on page 393.
3. For each hot spot, select or deselect the Correct option to specify whether the object is considered a correct or incorrect response when the user clicks it. You can select multiple correct answers.
4. You can delete the placeholder instances on the Stage. Place your movie clips on the Stage and use the Property inspector to give each movie clip the same instance name that you specified in the Component Inspector.

Adding and removing hot spot distractors

You can include from one to eight distractors (choices) in a Hot Spot learning interaction. You can change the default number of six distractors by adding more distractors or deleting existing distrators.

In general, you place the hot spot distractors over another graphic that the user is really intended to see. You might want to make your hot spot assets semi-invisible during authoring to visualize this effect. You can do this by turning the alpha effect setting down on each hot spot. The interaction scripts override this setting at runtime.

To add a hot spot distractor:

1. Create a movie clip symbol containing the graphics for the distractor object. For example, if you have an image that will have six hot spots, and you want to add a seventh choice, create a movie clip of the seventh graphic and place it in the library.
2. Select the Hot Spot component on the Stage, and then drag the symbol from the Library panel to the Stage.
3. In the Property inspector, name the instance. See "Naming and registering graphic distractors" on page 409.
4. Add the instance name to the Component inspector for the hot spot.

   The component does the rest of the work automatically at runtime.

To remove a hot spot distractor:

1. Select the hot spot instance that you want to remove, and delete it from the Stage.
2. Select the Hot Spot component (to the left of the Stage in the quiz template), and then display the Component inspector (Window > Development Panels > Component Inspector).
3. Remove the deleted object's instance name from the list in the Component Inspector.

Configuring a Multiple Choice interaction

In a Multiple Choice interaction, the user responds to a question with multiple answers; either one answer or several answers can be correct.

To configure a Multiple Choice interaction:

1. If you are not using a quiz template, place the learning interaction on the Stage. If you are using a quiz template, select the frame on the Interactions layer that contains the Multiple Choice interaction (Frame 6, if you have not added or removed keyframes).
2. Break the movie clip apart (Modify > Break Apart), display the Component Inspector, and then type the interaction ID and the question. See "Configuring a Learning Interaction component" on page 393.
3. Type the possible responses for the interaction (A–E).

   Note: You do not need to provide five responses. You can delete a response, but make sure to replace it or move any following responses up to the previous text box, if necessary, so that there are no blank text boxes between responses.

4. Select or deselect the Correct option to specify whether each response is considered correct or incorrect. You can have multiple correct answers.

Adding and removing multiple-choice distractors

You can include from one to eight distractors (choices) in a Multiple Choice learning interaction. You can change the default number of six distractors by adding more distractors or deleting existing distractors.

To add a multiple-choice distractor:

1. Select the frame with the Multiple Choice learning interaction in the Timeline.
2. Open the Flash UI Components folder in the Library panel (Window > Library) and drag a CheckBox component to the Stage.
3. In the Property inspector, name the instance. See "Naming and registering graphic distractors" on page 409.
4. Add the instance name to the Component inspector for the multiple-choice distractor.

   The component does the rest of the work automatically at runtime.

To remove a multiple-choice distractor:

1. Select the CheckBox instance that you want to remove, and delete it from the Stage.
2. Select the Multiple Choice component (to the left of the Stage in the quiz template), and then display the Component inspector (Window > Development Panels > Component Inspector).
3. Remove the deleted object's instance name from the list in the Component Inspector.

Configuring a True or False interaction

In a True or False interaction, the user responds with an answer of either True or False.

To configure a True or False interaction:

1. If you are not using a quiz template, place the learning interaction on the Stage. If you are using a quiz template, select the frame on the Interactions layer that contains the True or False interaction (Frame 7, if you have not added or removed keyframes).
2. Break the movie clip apart (Modify > Break Apart), display the Component Inspector, and then type the interaction ID and the question. See "Configuring a Learning Interaction component" on page 393.
3. In the Question text box, type the text of the question you want to ask the user.
4. Select Correct to specify which answer, True or False, is the correct response for the interaction. If you want, you can change these responses to Correct or Incorrect by changing the text of the distractors. For example, you could type **A. Correct** and **B. Incorrect** in the Distractors text boxes.

True or False interaction distractors

The True or False interaction includes a question text field, two RadioButton components, a control button, and a feedback text field. There are no other distractor options to configure.

Adding, naming, and registering assets

Each Flash learning interaction consists of the following assets:

- An interaction component
- Dynamic text fields
- Distractor elements
- User interface (UI) components

The collection of assets for each interaction type is stored within movie clip symbols in the library. These movie clips are intended to provide mobility for the assets so they can be copied to keyframes or among files. The movie clips are intended only to be containers and are not necessary to make the interaction work.

If you have experience handling and naming graphics, you can enter your own instance names for the graphic assets on the Stage. You do not need to use the movie clip containers or the templates—instead, you can add your own assets to the Stage, add a Learning Interaction component to the Stage, and then register the assets' instance names in the Component inspector for the interaction.

Remember the following aspects about naming assets:

- Interaction components do not need to be named.
- UI components need to have unique names for similar interaction types.
- Each graphic distractor (Drag object, Target object, hot spot, and hot object) must have a unique instance name.

- Text fields can share the same instance names across multiple interactions.

After you name the assets on the Stage, it's important to register those names in the Component inspector for the learning interaction so that the scripts can control the assets.

About naming Learning Interaction component instances

Every interaction has an interaction component associated with it to configure its unique parameters. These components do not need to be named.

Naming UI components (RadioButton, CheckBox, Button, and TextInput)

When you use similar interaction types, you need to give each UI component a unique name. For example, if you create two Multiple Choice interactions, the second interaction requires unique instance names for the CheckBox and the Button components. These new instance names need to be registered in the Component inspector for the learning interaction.

To name a UI component:

1. Select the UI component instance on the Stage.
2. In the Property inspector, type a name in the Instance Name text box.
3. Register the name in the Component inspector for the interaction (see "Registering dynamic text fields and UI components" on page 408).

Naming dynamic text fields

If you have more than one of any type of learning interaction in a quiz—for example, if you have two Drag and Drop learning interactions—the objects in each learning interaction must have unique names. These new unique instance names need to be registered in the Compoonent inspectorfor the learning interaction. See "Registering dynamic text fields and UI components" on page 408.

To name a dynamic text field:

1. Select the dynamic text field on the Stage.
2. In the Property inspector, type a name in the Instance Name text box.

   Note: Make sure to enter the instance name–not the variable name–in the Property inspector.

3. Register the name in the Component inspector (see the next section).

Registering dynamic text fields and UI components

After you enter the instance name for a dynamic text field or Button component in the Property inspector, you need to register the instance in the Component inspector for the interaction.

To register dynamic text fields and Button components:

1. Select the Learning Interaction component (to the left of the Stage in the quiz template), and open the Component Inspector, if necessary, from the Property inspector.
2. Click Assets, at the bottom of the panel.
3. Enter the name in the appropriate instance name text box.

Naming and registering graphic distractors

Graphic distractors such as Drag objects, Target objects, hot spots, and hot objects must be named uniquely across all interactions. This means that in a file with two Drag and Drop interactions, each containing four Drag objects, each of the eight Drag objects in the file must be named uniquely. For example, the Drag objects in the first interaction could be named Drag 1, Drag 2, Drag 3, and Drag 4, and the Drag objects in the second interaction could be named Drag A, Drag B, Drag C, and Drag D. This system ensures that the scripts work properly and the interactions behave as intended.

To name graphic distractors:

1. Make sure that the objects on the Stage are instances of learning interactions or movie clip symbols.
2. Select an object on the Stage—for example, a Target object.
3. In the Property inspector, type a name in the Instance Name text box.
4. Repeat steps 1–3 for each object on the Stage.
5. Register the names (see the following procedure).

Note: A sequential naming scheme is usually the easiest to work with—for example, Drag1, Drag2, Drag3, and so on.

To register a distractor instance name:

1. Select the Learning Interaction component (to the left of the Stage in the quiz template), and open the Component inspector from the Property inspector, if necessary.
2. Enter the name in the Component inspector, under Instance Name.

Text field names

Text fields can share the same names from interaction to interaction. That means that the question text field in interaction 1 can be named the same as the question text field in interaction 2, and so on. These names need to be registered with the interaction components, as do all assets names (see "Registering dynamic text fields and UI components" on page 408).

Asset name defaults

The assets supplied in the movie clip interaction containers are prenamed with the instance names listed in the following tables.

Drag and Drop learning interaction asset names

Asset	Description	Object type	Instance name
Question text field	Holds question text	Dynamic text field	Template_Question
Feedback text field	Holds feedback text	Dynamic text field	Template_Feedback
Control button	Submits user response and controls navigation	Flash UI Button component	Template_ControlButton

Asset	Description	Object type	Instance name
Reset button	Resets Drag objects	Flash UI Button component	Template_ResetButton
1-8 Drag objects	Drag object distractors	Movie clip symbol	Drag1 – Drag8
1-8 Target objects	Targets for Drag objects	Movie clip symbol	Target1 – Target8

Fill in the Blank learning interaction asset names

Asset	Description	Object type	Instance name
Question text field	Holds question text	Dynamic text field	Template_Question
Feedback text field	Holds feedback text	Dynamic text field	Template_Feedback
User entry field	User types answer into this text field	Flash UI TextInput component	Template_UserEntry
Control button	Submits user response and controls navigation	Flash UI Button component	Template_ControlButton

Hot Object learning interaction asset names

Asset	Description	Object type	Instance name
Question text field	Holds question text	Dynamic text field	Template_Question
Feedback text field	Holds feedback text	Dynamic text field	Template_Feedback
Control button	Submits user response and controls navigation	Flash UI Button component	Template_ControlButton
Reset button	Resets hot object distractors	Flash UI Button component	Template_ResetButton
1-8 hot objects	Hot object distractors	Movie clip symbol	HotObject1 - 8

Hot Spot learning interaction asset names

Asset	Description	Object type	Instance name
Question text field	Holds question text	Dynamic text field	Template_Question
Feedback text field	Holds feedback text	Dynamic text field	Template_Feedback
Control button	Submits user response and controls navigation	Flash UI Button component	Template_ControlButton
Reset button	Resets hot spot distractors	Flash UI Button component	Template_ResetButton
1-8 hot spots	Hot spot distractors	Movie clip symbol	HotSpot1 - 8

Multiple Choice learning interaction asset names

Asset	Description	Object type	Instance name
Question text field	Holds question text	Dynamic text field	Template_Question
Feedback text field	Holds feedback text	Dynamic text field	Template_Feedback
Control button	Submits user response and controls navigation	Flash UI Button component	Template_ControlButton
3-8 check boxes	Check box distractors	Flash UI CheckBox component	Checkbox1-8

True or False learning interaction asset names

Asset	Description	Object type	Instance name
Question text field	Holds question text	Dynamic text field	Template_Question
Feedback text field	Holds feedback text	Dynamic text field	Template_Feedback
Control button	Submits user response and controls navigation	Flash UI Button component	Template_ControlButton
2 radio buttons	True or false radio button distractors	Flash UI RadioButton component	Template_Radio1, Template_Radio2

Setting feedback options for a learning interaction

Feedback options control the text that the user sees before and while responding to an interaction.

To set feedback options for an interaction:

1. Select the interaction component (to the left of the Stage in the quiz template).
2. If the Component inspector is not already visible, open it from the Property inspector; and then click Options at the bottom of the panel.
3. Select Feedback if you want the interaction to present comments to users before and after they submit a response. Then, enter a comment for the following:
   - For Tries, enter the number of tries that a user is given to provide a correct response.
   - For Initial Feedback, enter the feedback that appears before the user has interacted with the quiz—for example, **Click an object and drag it to the matching object.**
   - For Correct Feedback, enter the feedback that appears if the user's response is correct—for example, **Yes, that is correct.**
   - For Incorrect Feedback, enter the feedback that appears if the user's response is incorrect and tries is set to 1—for example, **No, that is incorrect.**
   - For Additional Tries, enter the feedback that appears if the user's response is incorrect and tries is set to more than 1—for example, **No, that is incorrect. Try again.**

Note: Users are allowed one try only for the True or False learning interaction, so there is no Additional Tries field for that interaction.

Setting Knowledge Track options for a learning interaction

Knowledge Track is an automatic data-tracking feature that lets you transmit student performance data to a LMS, such as Lotus LearningSpace, or to other back-end tracking systems. Knowledge Track works with both AICC- and SCORM-compliant learning management systems. Knowledge Track captures and/or stores student information internal to the Flash application and transmits that data to an HTML page.

To successfully send data to a tracking system, you must embed the SWF file containing your learning interactions into an HTML page and select the HTML template in publish settings for either Flash with AICC Tracking or Flash with SCORM Tracking. To support an AICC-compliant LMS, the HTML that embeds the SWF file needs to be part of a frameset. See "Preparing Flash learning interactions for web hosting" on page 416.

The tracking data captured and transmitted by Knowledge Track is based on an industry standard for courseware-to-tracking-system communications, the AICC (Aviation Industry CBT Committee) specification version 2. This standard specifies the following data elements for each interaction.

You can set values for these data elements using the Component inspector for an interaction:

- InteractionID
- ObjectiveID
- Weighting

Other data elements are automatically set or calculated:

- Question Type
- Correct Response
- User Response
- Result
- Date/Time
- Latency

To set Knowledge Track options for an interaction:

1. Select the Learning Interaction component to the left of the Stage in the quiz template.
2. If the Component inspector is not already visible, open it from the Property inspector, and then click Options at the bottom of the panel.
3. Select Knowledge Track if you are using the learning interaction in a document created using a quiz template and you want the learning interaction to send data to a server-side learning management database.
4. Enter a name in the Objective ID text box to specify an objective for the interaction.

   This is an optional parameter. If the interaction is related to an objective that is set up in the LMS, enter that Objective ID in this text box. Tracking still works if you leave the Objective ID text box blank.

5. Specify the Weighting value for the interaction. The quiz templates use this parameter to calculate the score in the Results page. The default value is 1.

   Weighting indicates the relative importance of a question. You can enter any numeric value. If all learning interactions have a weight of 1, they are all scored equally. A weight of 2 counts twice as much as a weight of 1 and half as much as a weight of 4. For example, you can give advanced questions a weight of 3 and beginning-level questions a weight of 1.

Setting navigation options for a learning interaction

Documents created using the quiz templates have built-in navigation; make sure to turn Navigation off if you're using a quiz template. For documents that do not use the quiz template, you can set navigation options that display a Next Question button in your document.

To set navigation options for an interaction:

1. Select the Learning Interaction component to the left of the Stage in the quiz template.
2. If the Component inspector is not already visible, open it from the Property inspector, and then click Options at the bottom of the panel.
3. Under Navigation, specify how the interaction proceeds after the user submits a response for this interaction:
   - Select Off to disable navigation. Select this option if you are using the quiz templates, because the templates include their own navigation.
   - Select Next Button to require that the user click a Next button after submitting a response. In the GoTo Action field, select either Stop or Play. The Next button is a Button component that you can use with stand-alone interactions independent of the quiz template.

     If you want to navigate to a labeled frame instead of the next frame, enter a frame label in the GoTo Label text box.

     The default text for the Next button is Next Question. If you want to change the text, see "Setting control button labels for a learning interaction" on page 414.
   - Select Auto GoTo Next Frame to have the interaction proceed to the next frame after the user submits a response.

     If Feedback is deselected and Knowledge Track is selected, the Auto GoTo Next Frame feature can be enabled. This feature submits a score after evaluation and immediately navigates to the next frame for the next interaction.

     Note: If Feedback is selected or Knowledge Track is deselected, Auto GoTo Next Frame is reset to Next Button and an error message appears in the Output panel.

Setting control button labels for a learning interaction

All six types of interactions use an instance of the same control buttons: Check Answer, Submit, Next Question, and Reset. The only exception to this is the True/False interaction, which does not use a Reset button. You can change the label for the instance of each button using the Component Inspector.

To change the label for an instance of a control button:

1. Select the Learning Interaction component, to the left of the Stage in the quiz template.
2. If the Component inspector is not already visible, open it from the Property inspector, and then click Assets at the bottom of the panel.
3. Edit the label name under Control Button Labels.
4. Select Control > Test Movie to view the new labels on the buttons.

Tracking to AICC- or SCORM-compliant learning management systems

The Flash learning interactions and quiz templates allow easy communication with both AICC- and SCORM-compliant LMSs. The code built into both the Flash documents and the corresponding HTML/JavaScript files send properly formatted data to the LMS. The stand-alone interactions send question data, while the quiz templates track the score and time spent overall.

Because of differences between the two tracking standards (AICC and SCORM), there are differences in the compliance of the files created using the Flash learning interactions and the quiz templates.

To be SCORM-compliant, content must call an initialize command when it is first started, or before any other tracking commands are sent to the LMS. The Flash with SCORM HTML template was designed to initialize communication with a SCORM-compliant LMS when the file is loaded. It also sends a finish communication to the LMS when the file is unloaded, if the finish command wasn't explicitly sent previously.

The files created using both the Flash learning interactions and the quiz templates can send tracking data to an AICC- and SCORM-compliant LMS. Individual interactions do not send overall score and tracking data, but they can send interaction or question data.

Files created by using the quiz templates to comply with either AICC or SCORM standards do not read data from the LMS into the Flash file.

Overview of the communication for AICC- and SCORM-compliant content

The following overview shows what a student experiences when completing a quiz, along with hidden steps that are not exposed to the student.

AICC communication overview

When a student takes an AICC-compliant quiz, the following events occur:

1. The LMS is opened.
2. The student logs in to the LMS.
3. The student navigates through the course structure to find an assignable unit (AU). In this case, assume it's a Flash quiz, built using a Flash quiz template.
4. The student starts the Flash content (the quiz).
5. The content is located on a web server (for example, http://myserver/flashcontent.htm). To track properly, the Flash file needs to be embedded in the Flash AICC tracking frameset. See "Preparing Flash learning interactions for web hosting" on page 416.

   Note: Communication with the LMS, and data tracking, is not exposed to your user.

6. The LMS creates two parameters that are appended to the end of the URL: AICC_URL and AICC_SID. When the content is launched, the final URL looks something like the following:

   ```
   http://myserver/flashcontent.htm?AICC_URL=http://mylmsserver/
     trackingurl.asp&AICC_SID=12345
   ```

7. The student progresses through the quiz.
8. The Flash learning interaction sends the tracking data to the LMS through the HTML/JavaScript tracking files. The tracking data is sent when the student answers a question or progresses to the next page.

SCORM communication overview

When a student takes an SCORM-compliant quiz, the following events occur:

1. The LMS is initialized.
2. The student logs in to the LMS.
3. The student starts a quiz built using a Flash quiz template.
4. The content is embedded in the Flash/SCORM HTML template, which is opened in a SCORM-compliant frameset.

   Note: This is not exposed to the user.

   The LMS is responsible for creating the SCORM-compliant frameset, which includes all the necessary functions to communicate back to the LMS.
5. The student progresses through the quiz.
6. The Flash file sends the tracking data to the LMS through the HTML/JavaScript tracking files.

Preparing Flash learning interactions for web hosting

In order for web users to see your Flash application, you need to embed it into a web page. The steps to prepare AICC- and SCORM-compliant files for web hosting are slightly different and are covered in the following two sections.

Preparing an AICC-compliant learning interaction for web hosting

To send tracking data to an AICC-compliant LMS, you need to enable tracking for the quiz and then publish the Flash application using the Flash with AICC Tracking template. You must place the file generated by Flash on your web server in the same directory and modify the frameset file with the name of your quiz, and then place it on the web server with the HTML and SWF files. In addition, your LMS must be AICC-compliant and reference the frameset. This file is called `frameset.htm` by default.

To prepare an AICC-compliant file for web hosting:

1. Open the document in Flash.
2. Select File > Publish Settings.
3. In the Publish Settings dialog box that appears, make sure that (at least) both Flash (SWF) and HTML files are selected in the Formats panel.
4. Click the HTML tab at the top of the Publish Settings dialog box, and select the Flash with AICC Tracking template from the Template pop-up menu.
5. Click the Publish button, and close the dialog box.
6. Place the files produced by publishing the Flash file and any files linked (such as, MP3 or FLV) on the web server in the same directory.

   Additional files are created if Detect Flash Version is selected in the HTML tab of the Publish Settings dialog box. Make sure to copy all the HTML files to your web server but not the FLA file.
7. Open the Learning Extensions Srvr Files folder, which is located in the Flash MX 2004 program folder in the en/First Run/HTML/Learning Extensions folder. Copy the contents of this folder (frameset.htm, results.htm, and the scripts folder) to the same web server directory as the SWF file and the HTML file published in Flash.
8. Open the new copy of the frameset.htm file in a text editor.

   The following lines are found in the frameset.htm file:

```
<frameset frameborder="0" border="0" framespacing="0" rows="*,1">
<frame src="Untitled-1.htm" name="content" frameborder="0">
<frame src="results.htm" name="cmiresults" scrolling="0" frameborder="0">
```

9. In the second line, change `Untitled-1.htm` to the name of the HTML file you published in Flash (typically the HTML filename specified in the formats Tab of Publish Settings).

   The main file references any HTML files that were created in the publishing process. For example, if myQuiz.htm, myQuiz_content.htm, and myQuiz_alternate.htm were created by publishing the document, myQuiz.htm replaces Untitled-1.htm in the frameset.htm file. Then, myQuiz.htm calls myQuiz_content.htm and myQuiz_alternate.htm when necessary.

10. Start the LMS system (or create the AICC Course Descriptor Files) that references the frameset.htm file.

Preparing a SCORM-compliant learning interaction for web hosting

To send tracking data to a SCORM-compliant LMS, you must enable tracking for the quiz and publish the learning interaction using the Flash with SCORM Tracking template. In addition, you must place the files generated by Flash on your web server in the same directory.

To prepare a SCORM-compliant learning interaction for web hosting:

1. Open the document in Flash.
2. Select File > Publish Settings.
3. In the Publish Settings dialog box that appears, make sure that (at least) both Flash (SWF) and HTML are selected in the Formats panel.
4. Click the HTML tab at the top of the Publish Settings dialog box, and select the Flash with SCORM Tracking from the Template pop-up menu.
5. Click the Publish button, and close the dialog box.
6. Place the files produced by publishing the Flash file on the web server in the same directory.
7. Start the LMS system and reference the name of the HTML file. Make sure the LMS is set to launch the SCORM tracking frameset.

Extending learning interaction scripts

Note: The information in this section is intended for intermediate and advanced developers who want to extend the interaction capabilities.

The Flash learning interactions use an organized data structure to store and retrieve information about each interaction session. This data structure powers the evaluations and provides new possibilities for developers wanting to extend tracking features. You can use it to retrieve industry-compliant tracking data. This data structure is called the SessionArray.

Note: SessionArray and session are reserved keywords on the level where the interactions reside. Do not use these words as identifiers for other data.

Accessing cumulative tracking data through the SessionArray

The following overview shows how data is tracked through the SessionArray:

- When the Flash application is run, the first interaction component to load creates a new Array on the level of the interaction assets.
- The component then creates a new instance of the LToolBox global class in index0 of the Array. The instance of `LToolBox` is a storage place for all of the interaction's data. Data is set or retrieved from the instance by using predefined property names. See "Predefined property names" on page 418.
- When the Timeline moves to the second interaction, that interaction's component creates an instance of `LToolBox global class` in index1 of the SessionArray.

- When the Timeline moves to the third interaction, that interaction's component creates an instance of `LToolBox global class` in index2 of the SessionArray. It continues with index3, index4, and so on, until all interactions are in an index.
- At the end of a series of interactions, all the data processed during those interactions are available and organized.

Note: The SessionArray is used the same way in the stand-alone interactions and the quiz interactions.

Possible uses

This information is useful to developers who need to extend tracking or analysis of the interactions, including creating customized quiz environments and creating quizzes in a format different from that of the Flash quiz templates.

Tracking properties available in the SessionArray

The property names reference standard interaction tracking values for both AICC and SCORM LMSs. You can retrieve an interaction's properties by referencing its location in the following command:

```
SessionArray[n].[property_name]
```

For example, to reference the `interaction_id` value for interaction #1, you would use the following command:

```
SessionArray[0].interaction_id
```

To reference the result value for interaction #2, you would use the following command:

```
SessionArray[1].result
```

Predefined property names

The following table describes the predefined property names:

Property name	Description
`interaction_id`	Unique interaction name
`interaction_type`	Type of interaction
`objective_id`	Objective identification number
`weighting`	Weighting value for this interaction instance; some interactions can have more weight than others
`correct_response`	Formatted correct response returned from the user parameters
`student_response`	Formatted student response returned from the evaluation
`result`	Result of the evaluation
`latency`	Elapsed time during this interaction session
`dateStamp`	Date when the interaction occurs
`timeStamp`	Time when the interaction starts

All the methods and properties of the `LToolBox` global class are available within each SessionArray index.

Basic structure of the Learning Interaction scripts and components

Now that you know how the interaction data are stored and retrieved, here's a little more information to complete the picture. The Learning Interaction components are really the center of the e-learning setup. They collect user parameters and build the SessionArray and the interaction event handling functions on the level of the interaction assets. That is, they accept user parameters and configure the environment and assets accordingly. If you want to examine how these components work, you need to open the scripts in the Library panel.

Most of the scripts reside in one of two places. The first is the `LToolBoxglobalclass` script. This script processes data storage and data formatting for the interaction. The second script location is within each interaction component. These scripts initialize event handling functions triggered by the interaction assets. This is where the user parameters and interaction assets are initialized and the interaction evaluations scripts reside. Although these scripts are built on the component level, they are initialized on the same level as the interaction assets and submit data to the SessionArray on the interaction assets level.

To explore the scripts or add to them, look in the library for the 1_GlobalClass folder to access the `LGlobalClass` movie clip that contains the `LToolBoxglobalclass` script. Look in the 2_Components folders to access each Learning Interaction component script. Each script is split into commented sections that are described at the top of the script. Most of the script sections are built within functions for modularity.

Reviewing or editing the LToolboxClass script

The `LToolboxClass` script creates a built-in object that each interaction can use for data storage and basic functionality. The data pattern and functionality shared by all interactions is defined in this script. You can access the `LToolboxClass` script from the library.

To review or edit the LToolboxClass script:

1. In the Library panel, select Learning Interactions > Assets > Controls > ComponentSuperClass.
2. In the ComponentSuperClass folder, double-click the SuperClass movie clip to open it in symbol-editing mode.
3. In the Timeline for the movie clip, select Frame 1 and open the Actions panel, if necessary (Window > Development Panels > Actions).
4. Review or edit the script, as desired.

APPENDIX A
Using Samples and Templates

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 come with several samples and templates to help you get started. This appendix contains information on how to use samples and how to use templates.

Using samples

One of the quickest ways to learn Flash is to look at existing sample files to see how they were created. Each of the samples provided is described in the subsections below.

To view the published application file (SWF), you can link to it directly from the following pages describing each sample application. To view the design file (FLA) for a sample, open the file from within Flash. Some samples are complete applications, and others are simple applications that are intended to introduce a concept that you can use to build your own Flash content.

To open a sample FLA file:

1. In Flash, select File > Open.
2. Do one of the following to open the FLA file:
   - On Windows, browse to *boot drive*\Program Files\Macromedia\Flash MX 2004\Samples*sample folder,* and double-click *sample.fla*.
   - On Macintosh, browse to *Macintosh HD*/Applications/Macromedia Flash MX 2004/Samples/*sample folder,* and double-click *sample.fla*.

The Macromedia website contains more samples. You can view them at: www.macromedia.com/go/flashmx_samples.

Using the accessibility features in Flash

This sample shows how you can use the accessibility features of Flash. Features covered in this sample include tab ordering, components, and the Accessibility panel. In this sample, you can see how to use the new authoring tool features and user interface that are geared toward building applications containing the accessibility features. An arrow moves to indicate which element on the Stage has the focus. You can explore the source code to learn how to use the Flash accessibility features.

Building a photo scrapbook

This sample shows how to build an interactive photo scrapbook easily using behaviors instead of scripting. Behaviors provide an easy way to add interactivity to Flash content without having to write ActionScript. In this example, multiple behaviors are combined to create an interactive scrapbook. You can explore the source to learn more or to customize it to add your own pictures.

Customizing the Flash Player context menu

In this sample, you can see how to add custom options to the Flash Player context menu using ActionScript 2.0. This sample uses ActionScript to modify the context menu by adding cut, copy, and paste functionality that can be applied to the drawing objects on the Stage. The source includes the document file (FLA) and the ActionScript file (AS) that defines the ClipBoard class. You can explore both files to learn more about the context menu and writing classes with ActionScript 2.0.

Using device font masking

This sample explores the new Flash Player support for masking device fonts. The main features covered in the sample are device font masking, components, and scriptable masks. Newly added support in Flash Player for masking device fonts expands the possibilities for using scriptable masks on Flash content. Device font masking allows device fonts to be used inside components that mask their content, as well as custom masks that you create. This sample shows examples of both types of masking.

Developing multilingual content

This sample, featuring the Strings panel, shows a streamlined approach to developing and managing content in several languages. The new Strings panel provides for easy and fast content development in different languages. Localized content for text fields inside the document is kept in language-specific XML files in directories alongside the document. This example has content in several languages. The language that appears corresponds to the current language of the host operating system. You can explore this document to see how the Strings panel manages localized content.

Understanding text enhancements

This sample uses the many new text enhancements added to Flash MX 2004. The features highlighted in this sample are text styles, inline images, hypertext link improvements, and small text optimization. The text enhancements provide better and more precise control over text as it is entered in the Flash Player. This example loads an external HTML file named sample.html into a text field in the SWF file. However, you could use any text file that incorporates a tag-based format such as XML or HTML. New support for cascading style sheets lets Flash style the text in a given text field for each tag before displaying it. This example uses a style sheet named style.css. Additionally, Flash Player supports the `img` tag, which allows inline images that your text can wrap around.

Building custom panels with the Extensibility API

This sample covers how to design and build a panel to control the functions of Flash. The Trace Bitmap panel was built using the Extensibility API available in Flash. The Extensibility API is a series of JavaScript methods and properties that correspond to methods and properties inside the Flash application. You can explore the document to see how the JavaScript commands are used and to get ideas for building your own panel.

Building a news reader (Flash Professional only)

This sample provides an interface to read the latest news at DevNet on www.macromedia.com. The addition of the data binding user interface in Flash MX Professional 2004 lets users build interfaces that connect to, retrieve, and display remote data without writing any code. The new components have built-in data awareness, which allows several possible scenarios with web services, XML documents, and more. You can explore this sample to see how these components are connected to a Rich Site Summary (RSS) feed on www.macromedia.com.

Using scriptable masks

This sample covers how to dynamically mask Flash content at runtime using scriptable masks and components. Scriptable masking allows precise control over how masks behave at runtime, and allows the mask and/or the masked content to be dynamically changed at runtime. This sample shows different types of masks and masked content that are controlled by a user interface built with components. Explore this sample to learn more about masking and building interfaces with components.

Using advanced video features (Flash Professional only)

This sample demonstrates some of the more advanced video features available in Flash MX Professional 2004. Taking you through a computer generated world in FLV video, this application lets the user set video bookmarks and demonstrates advanced video techniques featuring non-linear video, executing functions from video cuepoints, and using ActionScript to pause, rewind, fast forward, restart and reposition video.

Also covered are the use and control of multiple MediaDisplay components, video synchronization, and skinning components.

Building a population viewer (Flash Professional only)

This sample demonstrates a unique and intuitive method of data visualization. A map of 15 European countries allows the user to drag the “population” of each country to either side of a scale. Human figures, representing the population of the country selected, are placed on the scale and automatically increase in size to represent the relative population of the country. Each time a country’s population is placed on the scale, the total population of each side of the scale is recalculated and the scale tips to the side with the greater total.

Web Services provide the initial country populations. Throughout the sample ActionScript is used to display, track, and compare data as well as operate the scale.

Using templates

Flash is equipped with several templates to help streamline your work. See the following sections for information about how to use each template:

To create a new document using a template:

1. Select File > New.
2. In the New Document dialog box, click the Template tab.
3. In the New from Template dialog box, select a Presentation template.
4. Add additional keyframes or screens to the presentation as needed.
5. If you add keyframes, make sure that all layers have the same number of keyframes.
6. Add your own content to the presentation.
7. Save and publish the file.

For specific information about how you can use a template, see the instructions for each template type.

Using advertising templates

Advertising templates facilitate the creation of standard rich media types and sizes defined by the Interactive Advertising Bureau (IAB) and accepted by the industry. For more information on IAB-endorsed ad types, see the IAB site at www.iab.net.

Testing with advertising templates

Ads should be tested for stability in a variety of browser and platform combinations. Your application is considered stable if it doesn't cause error messages, browser crashes, or system crashes.

Browser compatibility and requirements with advertising templates

You should work with webmasters and network administrators to create detailed testing plans that include tasks relevant to your specific users. These plans should be publicly available and updated regularly. Also, vendors should publish detailed plans indicating the browser and platform combinations in which their technologies are stable. Examples are available at the IAB Rich Media testing site at www.iab.net/standards/guidelines.asp. In addition, there might be additional requirements on size and file format of ads that vary by vendor and site. Check with your vendor, ISP, or the IAB to learn about these requirements that can affect the ad's design.

More information on rich media

The Macromedia Flash Advertising Alliance (MFAA) is an industry alliance focused on furthering Rich Media advertising and delivering great advertising experiences online. The MFAA offers a community discussion forum for advertising-related issues, technical resources for designers in the advertising space, and a list of voluntary guidelines for authors to guarantee the best possible Internet advertising experience.

Visit the MFAA and participate in the ongoing discussion at the Macromedia Flash Advertising Alliance website at www.mfaa.org.

Using video templates (Flash Professional only)

This section covers creating Flash content using video and includes instructions on using the video templates.

Flash MX Professional 2004 provides new ways to creatively use and deploy video in your Flash projects. The ability to play back external Flash Video (FLV) files enables authors to use video in more projects that can be viewable by wider audiences. The video templates provided with Flash Professional 2004 can help you create video presentations and user interfaces for selecting from multiple bandwidth-tuned streams of video.

Using the Bandwidth Selection template (Flash Professional only)

The Bandwidth Selection template uses forms and components to present the selection interface. This interface lets users control how much content they receive and lets authors tailor their applications to a variety of connection speeds. After the user makes a speed selection, the media playback component is directed to play the specified video.

The Select screen contains radio buttons that allow bandwidth selection. ActionScript to handle the selection of radio buttons is included within the Timeline of this screen.

To change the option labels or the number of options that users is presented, you can add, remove, or edit the components on the Select form.

Setting the URLs to the video content (Flash Professional only)

The media playback component progressively downloads FLV files without requiring them to be embedded in the SWF file.

The `data` property of the radio buttons is set to a string that is appended to a base string to properly form the correct URL. For example, if the user selects High Bandwidth and the base string is cartoon, the file that is loaded is cartoon_hi.flv.

To change the base string, open the Actions panel and select Frame 1 of the Actions layer of the Select screen. Edit the following ActionScript according to the instructions within the comments: <<Writer: do you need to add ("") after comments? MK>>

```
// Replace  "test" with your own base string. Be sure
// to keep the quotes.
var video_base:String = "test"
```

When the user makes a selection, the code appends the base string you've set with the string stored in the data property of the radio button, and the media playback component loads the media.

Using the Video Presentation template (Flash Professional only)

The Video Presentation template uses slides, media components, and behaviors to create a self-running presentation that progresses according to cues from the video playback. Video presentations are great for self-running demos, kiosks, or presentations to audiences over the web. At the end of the presentation, viewers have the option to play the presentation again from its beginning.

You can customize the presentation, add your own video and content, and customize the media playback component to broadcast events when you want.

Adding video (Flash Professional only)

The media display component on the Video slide handles the playback of the video in this presentation. To add video to the presentation, select the component on the Stage and replace the current value of the `URL` property with the URL of your media. Remember that after you publish, the SWF file always looks for the video at that location, so relative paths are recommended rather than hard-coded paths.

Setting cue points with the video templates (Flash Professional only)

Cue points are also set as properties of the media display component in the Parameters tab in the Component Inspector. You add new cue points to the list using the Add (+) button above the Cue Points list. Remove cue points with the Delete (-) button. Each cue point should be given a name and position.

If you give your cue point names and slides the same names, your presentation automatically navigates to the corresponding slide when a cue point is encountered.

Position is a point in time during the playback of the media file, starting from the beginning of the file, which is 0:0:0:0 (hours: minutes: seconds: frames/milliseconds). For example, to place a cue point 10 seconds into the file, enter **0:0:10:0**.

Adding content to the video templates (Flash Professional only)

Adding content to the video templates is as easy as adding new slides to the presentation and creating graphics and text, importing media, and adding animation. There are a few slides with some content to help get you started, but you can replace the content on the slides. After you've added content, you can use the Behaviors panel to add transitions between slides for eye-catching animation.

For more information on adding slides and transitions, see the instructions in "Using the screen presentation templates (Flash Professional only)" on page 429.

Using the Photo Slideshow template

The Photo Slideshow template lets you easily create and customize a photo slideshow.

Preparing your photos with the Photo Slideshow template

Photos must be in a suitable format to use the Photo Slideshow template. Flash lets you import images in a variety of formats, but JPEGs typically work best for photographs. For best results, save your photos as JPEGs using an image-editing program such as Macromedia Fireworks. Each image should have a size of 640 x 480 pixels and should be named in a numbered sequence. For example, for three files, the names could be photo1.jpg, photo2.jpg, and photo3.jpg.

Importing photos with the Photo Slideshow template

After your photo sequence is ready, you can import the sequence into a SWF file.

To import your files:

1. Select the layer of photos included in the example called Old Photos, and click the trash can icon to delete it.
2. Create a new layer by clicking the Insert Layer button, and name this new layer My Photos. Make sure that this new layer is the bottom layer.
3. Select the first blank keyframe in the My Photos layer, select File > Import, and locate your photo sequence.
4. Select the first image in the series, click Add, and click Import.
5. Flash recognizes that your image is part of a series and asks you to import all files in the series. Click Yes to complete the import process.

Adding finishing touches with the Photo Slideshow template

Flash places each image on separate keyframes. If you have more than four images, make sure that all the other layers have an equal number of frames. Your images appear in the Library panel. You can safely delete the old images that were included in this document from the library if you wish. Change the title, date, and caption at the top for each image. You can replace text as desired. You do not have to worry about the photo field. The template automatically determines how many images are in your document and indicates which photo you are currently using.

Using autoplay mode with the Photo Slideshow template

The Photo Slideshow template also has a built-in autoplay mode that automatically changes the photo after a set delay. The template is set to a default delay time of 4 seconds, but you can change this setting easily.

To adjust the delay:

1. Unlock the _controller layer.
2. Select the controller component.
3. Display the Parameters tab in the Component inspector by selecting Window > Development Panels > Component Inspector. The Parameters tab is selected by default.
4. Select the delay, and change this value to a new delay value in seconds.
5. Save and publish your document.

Using presentation templates

The presentation templates included with Flash let you create, customize, and publish your presentations.

Creating a slide presentation

Creating a slide presentation is as easy as adding new keyframes. Flash provides three slide layouts to get you started.

To create a slide presentation:

1. Select File > New.
2. In the New Document dialog box, click the Template tab.
3. In the New from Template dialog box, select a Presentation template.
4. In the Slide layer, add a keyframe for each slide in your presentation. For example, if your presentation has ten slides, add ten keyframes.
5. On each keyframe in the Slide layer, add the information you want to include in that slide. You can create or import graphics as well as add your company's logo, text, video, or audio to your presentation.
6. Make sure that all other layers have an equal number of frames.
7. Save and publish your document.

For more information on using the Timeline, see "Using the Timeline" in *Getting Started with Flash*.

Presenting your slides

Use the controls at the bottom of the application or your keyboard's arrow keys to move between slides during your presentation. Press the Left Arrow and Right Arrow keys to move to the corresponding previous and next slides; press the Up Arrow and Down Arrow keys to jump to the first and last slides.

You can also print each slide in your presentation by clicking the Print icon. If you know that you won't print your slides, you can delete the icon from the layout.

Customizing your slide presentation

If you want to change the colors in the template, select Modify > Movie, and change the background color. The presentation background changes to the newly selected color. Additionally, many templates come with alternate backgrounds. Show and hide the additional background layers to expose alternate designs.

You can match the background to your company's color scheme, or you can select something bright and eye-catching to capture your audience's attention.

Using the screen presentation templates (Flash Professional only)

The screen presentation templates included with Flash MX Professional 2004 use screens to make it easy for you to create a professional-looking slide presentation. You can add new slides to the outline and place text, graphics, imported media, and components on those slides to add to content.

After adding slides, you can use the Behaviors panel to add transitions between slides. Flash provides some sample slides with transitions to help you get started.

After you've customized your presentation, preview it by selecting Control > Test Movie from the application menu.

Slides have built-in navigation. Use the arrow keys on your keyboard or the navigation buttons that are part of the template's design to move forward and backward through your presentation.

Creating slides with the screen presentation templates (Flash Professional only)

The Screen Outline pane shows thumbnails of the slides that appear sequentially in your presentation. There are four ways to add new slides to a presentation.

To create a slide:

1. Create a new file using one of the screen presentation templates.
2. To add new slides to the presentation, do one of the following actions:
   - Select Insert > Screen.
   - Press Enter.
   - Click the Plus (+) button in the header of the Screen Outline pane.
   - Right-click to open the context menu, and select Insert Screen.
3. You can create slides that share graphical content such as logos by inserting nesting slides and placing the shared content on the parent slide. For example, the content that appears on the slide labeled Presentation appears on all the slides in the presentation. Insert nested screens by right-clicking in the Screen Outline pane and selecting Insert Nested Screen.

For more information on using slides and the outline pane, see Chapter 12, "Working with Screens (Flash Professional Only)," on page 215.

Adding transitions to the screen presentation templates (Flash Professional only)

After you've customized the content of your presentation, you can add animated transitions that help illustrate your points. Use the Behaviors panel to add transitions to your presentations.

To add transitions to a screen presentation:

1. Select the screen for which you'd like to add a transition.
2. If the Behaviors panel isn't visible, select Window > Development Panels > Behaviors.
3. Click the Add button (+) in the Behaviors panel, and select Screen > Set Transition.
4. Customize your transition in the dialog box. For information about each available transition style, see "Creating controls and transitions for screens with behaviors (Flash Professional only)" on page 228.
5. Click OK after you finish designing your transition.
6. Select the event on which you'd like your transition to start. The most common events for slide transitions are `onShow`, when the slide becomes visible, or `onHide`, when the slide is hidden.

For more information on behaviors, see "Controlling instances with behaviors" on page 65.

Using the mobile device templates

Flash content is viewable across multiple browsers, platforms, and mobile phones. You can author the following:

- High-quality animations
- Games
- Rich-media custom user interfaces for devices and desktop systems
- Immersive e-commerce and business solutions

Flash files are compact, which makes them perfect for wireless carrier networks, where transfer rates range between 9.6 and 60 kilobytes per second (Kbps). Mobile devices, unlike desktop computers, have limited storage capability, so the small footprint of Flash is ideal.

The mobile device templates let you create content for many mobile devices. Use the device skins in the templates to preview your content as it will look on the device.

Note: The skins are on guide layers and won't export with your content or appear at runtime.

For more information on authoring Flash files for mobile devices, see the Macromedia Mobile Devices site at www.macromedia.com/devnet/devices/.

Using quiz templates

You can use the quiz templates to create self-scoring quizzes with several interaction types. For information about using the quiz templates, see Chapter 19, "Creating E-learning Content," on page 387.

Using form application templates (Flash Professional only)

Flash MX Professional 2004 provides two templates you can use to create form-based applications:

- "Query-Error-Response template (Flash Professional only)" on page 431
- "Windowed Application template (Flash Professional only)" on page 432

Query-Error-Response template (Flash Professional only)

The Query-Error-Response template helps you create applications that perform a simple query to a remote data source, and then, depending on the outcome, display the results in a response form or show an error on an error form. This type of application is useful when performing queries on web services because they are structured as a simple query/response transaction. There are two steps for using this template, which are discussed in this section.

Configuring your service

The first step is to configure the service that your application will call. The template uses a web service connector. If you use a web service as your data source, you can configure the web service connector in the Parameters tab on the Component Inspector. Enter the URL to the service in the WSDL field, and select the operation your application will call.

You can replace the web service connector with another connector that is appropriate to your application from the Components panel. If you select your own connector, you can delete the web service component, but you must edit the actions in Frame 1 of the Application form to replace "wsc" with the instance name of the connector you created. This ensures that the Submit button triggers your service.

For more information about web service and other connectors, see Chapter 1, "About Components" in *Using Components*.

Customizing your forms

The next step is to customize your forms. The Query form should contain fields that correspond to the parameters of your service. The Response form should contain the fields that correspond to the results of your service. The Error form shows an error message to the user when something has gone wrong in the process of calling the service. You can show any message on the Error screen.

To customize the Query form:

1. Select the Query form in the Screen Outline pane.
2. Use components from the Components panel, such as text input fields, radio buttons, combo boxes, and others, to create the input fields for the Query form.
3. After you've laid out your form elements, use the Parameters tab on the Component Inspector to create bindings between your components and the parameters of your service connector.

To customize the Response form:

1. Select the Response form in the Screen Outline pane.
2. Use components to create fields that will show the results.

   For example, if your service is a weather service that returns temperature, you could use a label component to create a non-editable text display.

   After you've laid out your components, use the Parameters tab on the Component inspector to create bindings between your components and the results of your service connector.

To customize the Error form:

1. Select the Applications form. The Error form is shown using ActionScript in Frame 1 of the Applications form, during processing of the service call. The following example shows two event handlers:

```
function status (stat) {
  // Handle status message for errors
  // If error,
  // showError();
}

function result (res) {
  // Handle result message for errors
  // If error,
  // showError();
}
```

2. You can replace the bodies of these functions with your own code to interpret the status and result messages and do one of the following actions:
   - Catch an error and show the error screen.
   - Go to the results screen to show the service's response.

To learn more about the result and status messages of a service call, see "Data Integration (Flash Professional Only)" on page 253.

Windowed Application template (Flash Professional only)

The Windowed Application template helps you create a windowed application that consists of layered content panes. These panes are draggable, and they rise to the topmost layer when the focus is on them. Each window can contain different content for the user to interact with.

The window components that load the subforms are on the Application form. The `contentPath` property of each window component corresponds to the instance name of the form that it will load at runtime.

Modifying and adding window content

Window content is created on subforms of the Application form. The template comes with four forms: a calendar, a DataGrid component showing simulated inbox content, a scroll pane showing an image, and a login form.

To modify window content:

1. Select any form in the outline and replace the contents with components of your choice. You can even add data connectors to populate your components with remote data. For more information about data connectors, see Chapter 1, "About Components" in *Using Components*.
2. After altering the contents of a form, make sure that the window component that will load the form is sized properly, so your form's content won't appear clipped at runtime.

To add new windows and content:

1. Create a new form in the Screen Outline pane, and give it an instance name. Make sure that its `visible` property is set to `false`.
2. Create a new window component on the Application screen, and set its `contentPath` property to the instance name of the form you've created.
3. Add content to your new form.

   At runtime, a copy of your form is loaded into the window component.

APPENDIX B
XML to UI

Macromedia Flash MX 2004 and Macromedia Flash MX Professional 2004 come with several extensibility features including Behaviors, Commands (JavaScript API), Effects, and Tools. With these features, advanced users can extend or automate the authoring tool's functionality. The XML to UI engine works with each of these extensibility features to create dialog boxes that the user sees if the extension either requires or accepts parameters.

XML to UI uses a subset of the XML User Interface Language (XUL), along with some tags created for Flash. These tags define a dialog box exclusively with XML. The XML to UI rendering engine parses the XML and generates a *modal* dialog box. Modal dialog boxes, unlike *modeless* dialog boxes, must be dismissed (either accepted or cancelled) before the application can continue operation.

When used with Behaviors, the XML tags that define the dialog box reside in the same XML file in which the behavior is defined. For Effects, Tools, and the JavaScript API, the XML tags must be placed in a separate XML file.

Layout tag summary for XML to UI dialog boxes

The following tags are used for dialog box layout:

Tag	Description
`<column>`	Creates one column in a tabular grid layout.
`<columns>`	Creates a container tag for the `<column>` tags in a tabular grid layout.
`<dialog>`	Creates the container tag for the entire dialog box.
`<grid>`	Creates a container for tabular layout using <rows> and <columns>.
`<hbox>`	Creates a container for items laid out horizontally.
`<row>`	Creates one row in a tabular grid layout
`<rows>`	Creates a container tag for the `<row>` tags in a tabular grid layout.
`<separator>`	Creates a separator bar that displays vertically in an <hbox> and horizontally in a <vbox>.

Tag	Description
<spacer>	Creates a transparent fill space used to arrange controls.
<vbox>	Creates a container for items laid out vertically.

Control tag summary for XML to UI dialog boxes

The following XML tags are used to create controls:

Tag	Description
<button>	Creates a button control.
<checkbox>	Creates a check box control.
<choosefile>	Creates a file chooser control (this is not part of the XUL standard).
<colorchip>	Creates a color picker control (this is not part of the XUL standard).
<flash>	Creates a container for an embedded SWF file (this is not part of the XUL standard).
<label>	Creates a text label that can be associated with another control.
<listbox>	Creates a list box control to contain ‹listitem› tags.
<listitem>	Creates an individual item in a list box control.
<menulist>	Creates a pop-up menu control that contains ‹menupop› and ‹menuitem› tags.
<menupop>	Creates the pop-up menu in a pop-up menu control; contains ‹menuitem› tags.
<menuitem>	Creates an individual item in a pop-up menu control.
<popupslider>	Creates a pop-up slider control (this is not part of the XUL standard).
<property>	Creates a custom property in an embedded SWF file; used with the ‹flash› tag.
<radiogroup>	Creates a container for a group of radio button controls.
<radio>	Creates a single radio button control. This tag must be used within a ‹radiogroup› tag.
<targetlist>	Creates a control that lists all instances of a class and lets the user select an instance.
<textbox>	Creates a control that allows the input of text.

‹column›

Availability

Flash MX 2004.

Usage

```
<column>
   ...
   child tags
   ...
</column>
```

Attributes

None.

Child tags

Control tags.

Parent tag

<columns>

Description

Layout tag; creates one column in a tabular grid layout. The column tag must be within a <columns> tag, which must be within a <grid> tag.

Example

See the example for <grid>.

<columns>

Availability

Flash MX 2004.

Usage

```
<columns>
   ...
   child tags
   ...
</columns>
```

Attributes

None.

Child tags

<column>

Parent tag

<grid>

Description

Layout tag; creates a container tag for the <column> tags in a tabular grid layout. The <columns> tag must be within a <grid> tag.

Example

See the example for <grid>.

<dialog>

Availability

Flash MX 2004.

Usage

```
<dialog
  id = "myID"
  title="yourTitle"
  buttons="accept[, cancel]">
...
child tags
...
</dialog>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the dialog box and access the values it returns.

`title` String; text that appears in the title bar of the dialog box.

`buttons` Accepts either or both of the strings "accept" and "cancel", which represent the OK and Cancel buttons, respectively.

Child tags

<hbox>,<grid>,<vbox>

Description

Layout tag; creates the container tag for the entire dialog box. All other tags used must be contained in this tag.

Example

For an example that uses the <dialog> tag with the <hbox> and <vbox> tags, see the examples in <hbox> and <vbox>. For an example that uses the <dialog> tag with the <grid> tag, see the example in <grid>.

<grid>

Availability

Flash MX 2004.

Usage

```
<grid>
  ...
  child tags
  ...
</grid>
```

Attributes

None.

Child tags

<columns>, <rows>

Parent tag

<dialog>

Description

Layout tag; creates a container for tabular layout using <rows> and <columns> tags.

Example

The following example uses the <grid>, <columns> and <rows> tags to define a dialog box. To see how this dialog box works with a JavaScript API command, see the example for <menulist>.

```
<dialog id="scale-dialog" title="Scale Selection" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="center">
        <label value="Scale x: " control="xScale"/>
        <textbox id="xScale"/>
      </row>
      <row align="center">
        <label value="Scale y:" control="yScale"/>
        <textbox id="yScale" />
      </row>
    </rows>
  </grid>
</dialog>
```

<hbox>

Availability

Flash MX 2004.

Usage

```
<hbox>
  ...
  child tags
  ...
</hbox>
```

Attributes

None.

Child tags

<hbox>, <vbox>

Parent tag

<dialog>, <hbox>, <vbox>

Description

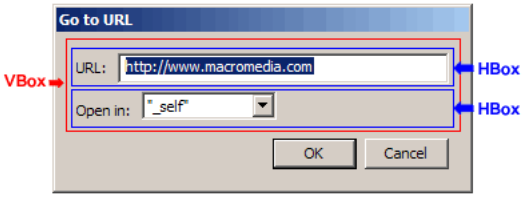
Layout tag; creates a container for items laid out horizontally. All layout objects defined within <hbox> tags are arranged horizontally with respect to one another. By default, each layout object is evenly spaced, but this setting can be altered with the <space> tag.

Example

The following example is excerpted from a Behaviors definition file, Web_Goto_Webpage.xml, and defines a dialog box with a textbox control and a drop-down menu control:

```
<dialog id="GotoWebPage-dialog" title="Go to URL" buttons="accept, cancel">
  <vbox>
    <hbox>
      <label value="URL:" control="URL"/>
      <textbox literal="true" required="true" width="40" id="URL"/>
    </hbox>
    <hbox>
      <label value="Open in:" control="targetWindow"/>
      <menulist literal="true" id="targetWindow">
        <menupopup>
          <menuitem label='"_self"'/>
          <menuitem label='"_parent"'/>
          <menuitem label='"_blank"'/>
          <menuitem label='"_top"'/>
        </menupopup>
      </menulist>
    </hbox>
  </vbox>
</dialog>
```

The following graphic shows the Go to URL dialog box. Red and blue outlines of the VBox and HBox containers have been added to show how these container tags are used to define the layout:



<row>

Availability

Flash MX 2004.

Usage

<row>

```
  ...
  child tags
  ...
</row>
```

Attributes

None.

Child tags

Control tags.

Parent tag

<rows>

Description

Layout tag; creates one row in a tabular grid layout. The row tag must be within a <rows> tag, which must be within a <grid> tag.

Example

The following example uses the <row> tag to help define a dialog box. To see how this dialog box works with a JavaScript API command, see the example for <menulist>.

```
<dialog id="scale-dialog" title="Scale Selection" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="center">
        <label value="Scale x: " control="xScale"/>
        <textbox id="xScale"/>
      </row>
      <row align="center">
        <label value="Scale y:" control="yScale"/>
        <textbox id="yScale" />
      </row>
    </rows>
  </grid>
</dialog>
```

<rows>

Availability

Flash MX 2004.

Usage

```
<rows>
  ...
  child tags
```

```
  ...
</rows>
```

Attributes

None.

Child tags

<row>

Parent tag

<grid>

Description

Layout tag; creates a container tag for the row tags in a tabular grid layout. The columns tag must be within a <grid> tag.

Example

The following example uses the <grid>, <columns> and <rows> tags to define a dialog box. To see how this dialog box works with a JavaScript API command, see the example for <menulist>.

```
<dialog id="scale-dialog" title="Scale Selection" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="center">
        <label value="Scale x: " control="xScale"/>
        <textbox id="xScale"/>
      </row>
      <row align="center">
        <label value="Scale y:" control="yScale"/>
        <textbox id="yScale" />
      </row>
    </rows>
  </grid>
</dialog>
```

<separator>

Availability

Flash MX 2004.

Usage

```
<separator/>
```

Attributes

None.

Child tags

None.

Parent tag

<hbox>,<vbox>

Description

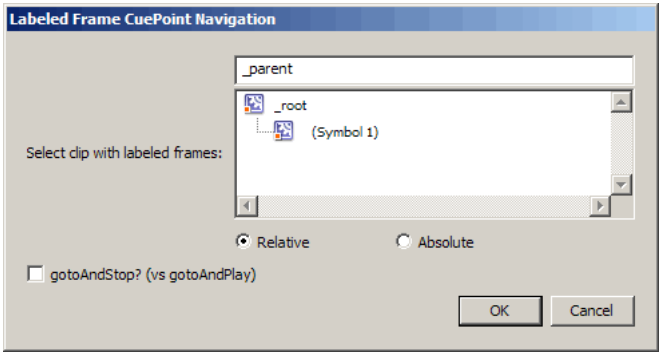
Layout tag; creates a separator bar that displays vertically in an <hbox> and horizontally in a <vbox>.

Example

The following example adds a separator bar to the Labeled Frame Cuepoint Navigation behavior dialog box. The current dialog box, which comes with Flash MX 2004, is defined in the file CuePointNamedFrame.xml.

```
<dialog id="NamedFrameCuePointDialog" title="Labeled Frame CuePoint
  Navigation" buttons="accept, cancel">
  <vbox>
    <hbox>
      <label value="Select clip with labeled frames:" control="target"
  required="true" />
      <targetlist id="target" class="movieclip" />
    </hbox>
    <hbox>
      <checkbox id="stop" label="gotoAndStop? (vs gotoAndPlay)"
  checked="false" />
    </hbox>
  </vbox>
</dialog>
```

These tags produce the following dialog box:

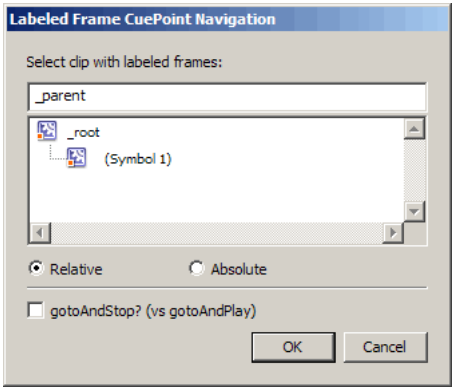


The following example adds a separator bar and removes the <hbox> tags:

```
<dialog id="NamedFrameCuePointDialog" title="Labeled Frame CuePoint
  Navigation" buttons="accept, cancel">
  <vbox>
    <label value="Select clip with labeled frames:" control="target"
  required="true" />
```

```
    <targetlist id="target" class="movieclip" />
    <separator/>
    <checkbox id="stop" label="gotoAndStop? (vs gotoAndPlay)" checked="false"
  />
  </vbox>
</dialog>
```

The modified tags produce the following dialog box:



<spacer>

Availability

Flash MX 2004.

Usage

```
<spacer/>
```

Attributes

None.

Child tags

None.

Parent tag

<column>,<hbox>,<row>,<vbox>

Description

Layout tag; creates a transparent fill space used to arrange controls.

Example

The following example uses the JavaScript API to create a simple command that sends selected values to the Output Panel. Create the files, as described in this section, and place them in the Commands folder in your user-level configuration folder.

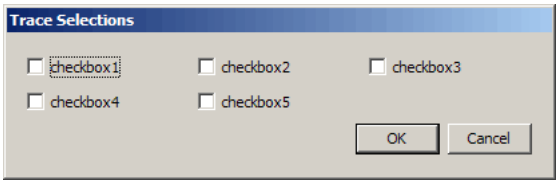
Create a file named Trace Selections.jsfl. Place the following code into the file and save the file:

```
// Create an XML to UI dialog box using the XML definition in the
// Trace Selections.xml file
var traceSelectionsDlg = fl.getDocumentDOM().xmlPanel( fl.configURI +
  "Commands/Trace Selections.xml" );

if (traceSelectionsDlg.dismiss == "accept") {
  fl.trace("Checkbox 1: " + traceSelectionsDlg.checkbox1);
  fl.trace("Checkbox 2: " + traceSelectionsDlg.checkbox2);
  fl.trace("Checkbox 3: " + traceSelectionsDlg.checkbox3);
  fl.trace("Checkbox 4: " + traceSelectionsDlg.checkbox4);
  fl.trace("Checkbox 5: " + traceSelectionsDlg.checkbox5);
}
```

Next, create a file named Trace Selections.xml (this example does not use the `<spacer/>` tag, so the second row of check box controls are aligned on the left). Place the following code into the file and save the file:

```
<dialog id="traceSelections" title="Trace Selections" buttons="accept,
  cancel">
  <vbox>
    <hbox>
      <checkbox id="checkbox1" label="checkbox1"/>
      <checkbox id="checkbox2" label="checkbox2"/>
      <checkbox id="checkbox3" label="checkbox3"/>
    </hbox>
    <hbox>
      <checkbox id="checkbox4" label="checkbox4"/>
      <checkbox id="checkbox5" label="checkbox5"/>
    </hbox>
  </vbox>
</dialog>
```

The `Trace Selections` command now appears on the Commands menu. If you select the `Trace Selections` command from the Commands menu, the dialog box defined by Trace Selections.xml appears, as shown in the following figure:

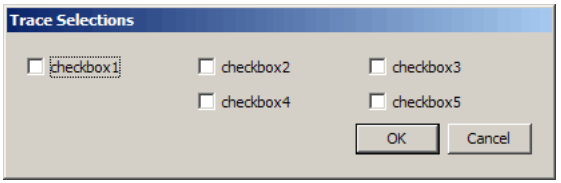


Finally, add the `<spacer/>` tag to the Trace Selections.xml file:

```
<dialog id="traceSelections" title="Trace Selections" buttons="accept,
  cancel">
  <vbox>
    <hbox>
      <checkbox id="checkbox1" label="checkbox1"/>
      <checkbox id="checkbox2" label="checkbox2"/>
      <checkbox id="checkbox3" label="checkbox3"/>
    </hbox>
    <hbox>
      <spacer/>
```

```
      <checkbox id="checkbox4" label="checkbox4"/>
      <checkbox id="checkbox5" label="checkbox5"/>
    </hbox>
  </vbox>
</dialog>
```

Adding the `<spacer/>` tag to the second row of check boxes pushes checkbox4 and checkbox5 to the right:



<vbox>

Availability

Flash MX 2004.

Usage

```
<vbox>
  ...
  child tags
  ...
</vbox>
```

Attributes

None.

Child tags

`<hbox>`, `<vbox>`, control tags

Parent tag

`<grid>`

Description

Layout tag; creates a container for items laid out vertically.

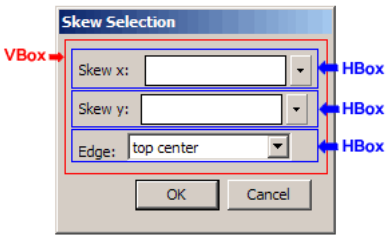
Example

The following example redefines the `<grid>`-based dialog box used in the `<popupslider>` example with `<vbox>` and `<hbox>` tags instead:

```
<dialog id="skew-dialog" title="Skew Selection" buttons="accept, cancel">
  <vbox>
    <hbox>
      <label value="Skew x: " control="xSkew" align="left"/>
      <popupslider id="xSkew" minvalue="-180" maxvalue="180"/>
    </hbox>
    <hbox>
```

```
      <label value="Skew y:" control="ySkew" align="left"/>
      <popupslider id="ySkew" minvalue="-180" maxvalue="180"/>
    </hbox>
    <hbox>
      <label value="Edge:" control="edge" align="left"/>
      <menulist id="edge">
        <menupop>
          <menuitem label="top center"/>
          <menuitem label="right center"/>
          <menuitem label="bottom center"/>
          <menuitem label="left center"/>
        </menupop>
      </menulist>
    </hbox>
  </vbox>
</dialog>
```

The following figure shows the Skew Selection dialog box defined using `<vbox>` and `<hbox>` tags instead of the `<grid>` tag. Red and blue outlines of the VBox and HBox containers have been added to show how these container tags are used to define the layout:



‹button›

Availability

Flash MX 2004.

Usage

```
<button
  id="myID"
  label="myLabel"
  tabindex="myIdx"
  accesskey="myChar"/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`label` String; text that appears on the button.

`tabindex` Number; an integer used to set the control's position in the tab order (available only on Windows).

`accesskey` String; a character to be used for the keyboard shortcut for this control (available only on Windows).

`oncommand` A JavaScript command that executes when the button is clicked.

Child tags

None.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a button control.

Example

The following example uses the JavaScript API to create a new command that appears in Commands menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

First, create a file named button.jsfl and place it in your Commands folder. Place the following code into the file and save the file:

```
// Create an XML to UI dialog box using the XML definition
// in the button.xml file
var buttonDlg = fl.getDocumentDOM().xmlPanel( fl.configURI + "Commands/
  button.xml" );
```

Second, create a file named button.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<?xml version="1.0"?>
<dialog id="button-dialog" title="Button Example" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
    </columns>
    <rows>
      <row>
        <label width="150" value="The following buttons will send text to the
  Output Panel"/>
      </row>
      <row>
        <button id="helloBtn"  label="Hello" oncommand="fl.trace('Hello')"/>
      </row>
      <row>
        <button id="worldBtn" label="world" oncommand="fl.trace('world')"/>
      </row>
    </rows>
```

```
  </grid>
</dialog>
```

The `button` command now appears on the Commands menu. If you have a Flash document open, you can select the `button` command from the Commands menu and the dialog box defined by button.xml appears.

<checkbox>

Availability

Flash MX 2004.

Usage

```
<checkbox
  id="myID"
  label="myLabel"
  tabindex="myIdx"
  checked="true|false"
  accesskey="myChar"/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`label` String; text that appears next to the check box.

`tabindex` Number; an integer used to set the control's position in the tab order (available only on Windows).

`checked` Boolean value; set the default value. If `true`, the box is checked when the dialog box first appears; `false` otherwise.

`accesskey` String; a character to be used for the keyboard shortcut for this control (available only on Windows).

Child tags

None.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a check box control.

Example

The following example is excerpted from the Labeled Frame CuePoint Navigation behavior definition file:

```
<dialog id="NamedFrameCuePointDialog" title="Labeled Frame CuePoint
  Navigation" buttons="accept, cancel">
  <vbox>
    <hbox>
```

```
      <label value="Select clip with labeled frames:" control="target"/>
      <targetlist id="target" class="movieclip" />
    </hbox>
    <hbox>
      <checkbox id="stop" label="gotoAndStop? (vs gotoAndPlay)"
  checked="false" />
    </hbox>
  </vbox>
</dialog>
```

<choosefile>

Availability

Flash MX 2004.

Usage

```
<choosefile
  id="myID"
  literal="true|false"
  pathtype="relative|absolute"
  required="true|false"
  size="mySize"
  tabindex="myIdx"
  type="open|save"
  value="myValue"
  width="myWidth"/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`literal` Boolean value; if `true` then the value returned from this control is enclosed by quotation marks (""). If `false`, which is the default setting, the returned value does not have quotation marks.("").

`pathtype` String; the two possible values are `relative` and `absolute`.

`required` Boolean value: if `true`, then the OK button cannot function until a value is entered for this control; if `false`, the control has no effect on the OK button.

`size` Number; an integer that sets the width of the input field using the average character width.

`tabindex` Number; an integer used to set the control's position in the tab order (available only on Windows).

`type` String; can be either `"open"` or `"save"`.

`value` String; default text that appears in the text input area.

`width` Number; sets the width of the text input area measured in pixels.

Child tags

None.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a file chooser control (this is not part of the XUL standard). This control provides users with access to the operating system's file selection dialog box.

Example

The following example uses the JavaScript API to create a new command that appears in Commands menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

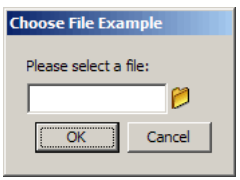
First, create a file named choosefile.jsfl and place it in your Commands folder. Place the following code into the file and save the file:

```
// Create an XML to UI dialog box using the XML definition
// in the choosefile.xml file
var chooseFileDlg = fl.getDocumentDOM().xmlPanel( fl.configURI + "Commands/
  choosefile.xml" );
if (chooseFileDlg.dismiss == "accept") {
  var path = chooseFileDlg.choosefileControl;
  fl.trace(path);
}
```

Second, create a file named choosefile.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<?xml version="1.0"?>
<dialog id="choosefile-dialog" title="Choose File Example" buttons="accept,
  cancel">
  <vbox>
    <label value="Please select a file: "/>
    <choosefile id="choosefileControl" type="open" pathtype="relative"/>
  </vbox>
</dialog>
```

The `choosefile` command now appears on the Commands menu. If you have a Flash document open, you can select the `choosefile` command from the Commands menu and the dialog box defined by choosefile.xml appears, as shown in the following figure:



<colorchip>

Availability

Flash MX 2004.

Usage

```
<colorchip
  id="myID"
  color="myColor"/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`color` Number; hex number representing a color used as default value.

Child tags

None.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; Creates a color picker control (this is not part of the XUL standard). This tag is specific to Flash and is not a part of the XUL tag set.

Example

The following example uses the JavaScript API to create a new command that appears on the Commands menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

First, create a file named setcolor.jsfl and place it in your Commands folder. Place the following code into the file and save the file:

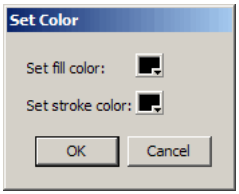
```
// Create an XML to UI dialog box using the XML definition
// in the setcolor.xml file
var setcolorDlg = fl.getDocumentDOM().xmlPanel( fl.configURI + "Commands/
  setcolor.xml" );

if (setcolorDlg.dismiss == "accept") {
  fl.getDocumentDOM().setFillColor(setcolorDlg.fillColor);
  fl.getDocumentDOM().setStrokeColor(setcolorDlg.strokeColor);
}
```

Second, create a file named setcolor.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<dialog id="setcolor-dialog" title="Set Color" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
```

```
      <column/>
    </columns>
    <rows>
      <row align="left">
        <label value="Set fill color: " control="fillColor" align="left"/>
        <colorchip id="fillColor" color="#000000"/>
      </row>
      <row align="left">
        <label value="Set stroke color:" control="strokeColor" align="left"/>
        <colorchip id="strokeColor" color="#000000"/>
      </row>
    </rows>
  </grid>
</dialog>
```

The `setcolor` command now appears on the Commands menu. If you have a Flash document open, draw a shape on the stage and select the `setcolor` command from the Commands menu. The dialog box defined by setcolor.xml appears, as shown in the following figure:



‹flash›

Availability

Flash MX 2004.

Usage

```
<flash
  id="myID"
  width="x"
  height="y"
  src="SWF file">
  ...
  child tags
  ...
</flash>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`width` Number; sets the width of the `<flash>` control measured in pixels.

`height` Number; sets the height of the `<flash>` control measured in pixels.

`src` String; path to the SWF file to be embedded in the dialog box.

Child tags

`<property>`

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a container for an embedded SWF file (this is not part of the XUL standard). The xmlui object in the JavaScript API allows getting and setting of parameter values in the embedded SWF file.

Example

The following example is excerpted from the blur.xml file, which defines the dialog box for the Blur Timeline Effect.

```
<dialog id="blur-dialog" title="Blur">
  <flash id="blur_ui" src="blur.swf" width="772" height="456">
    <property id="first" />
    <property id="dur" />
    <property id="hor" />
    <property id="vert" />
    <property id="regPoint" />
    <property id="blur_amount" />
    <property id="baseScale" />
  </flash>
</dialog>
```

<label>

Availability

Flash MX 2004.

Usage

```
<label
  control="myControlID"
  accesskey="char"
  value="myText"
  align="left|center|right"/>
```

Attributes

`control` String; the string identifier that matches the ID value of an associated control.

`accesskey` String; a character to be used for the keyboard shortcut for this control (available only on Windows).

`align` String; `left`, `center`, or `right` determines whether the text is aligned on the left, center or right, respectively.

`value` String; the text that appears in the dialog box.

Child tags

None.

Parent tag

<dialog>, <hbox>, <row>, <vbox>

Description

Control tag; creates a text label that can be associated with another control.

Example

The following example is excerpted from the HideScreen.xml file, which defines the Hide Screen Behavior.

```
<dialog id="SelectScreenDialog" title="Select Screen" buttons="accept,
  cancel">
  <vbox>
    <hbox>
      <label value="Select Screen:" control="TARGET"/>
      <targetlist id="TARGET" class="screen" />
    </hbox>
  </vbox>
</dialog>
```

<listbox>

Availability

Flash MX 2004.

Usage

```
<listbox
  id="myID"
  tabindex="myIdx"
  rows="numRows">
  ...
  child tags
  ...
</listbox>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`rows` Number; an integer representing the number of rows to display in the listbox.

`tabindex` Number; an integer used to set the control's position in the tab order (available only on Windows).

Child tags

<listitem>.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a listbox control to contain `<listitem>` tags.

Example

The following example modifies the `skew` command example from `<popupslider>` so that it uses a `<listbox>` control instead of a `<menulist>` control for the `edge` parameter to the JavaScript skewSelection() method.

The example uses the JavaScript API to create a new command that appears on the Commands menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

First, create a file named skewlist.jsfl and place it in your Commands folder. Place the following code into the file and save the file:

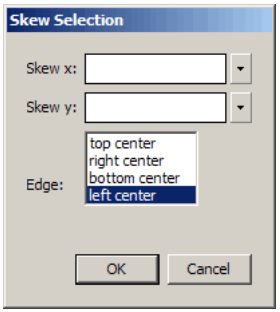
```
// Create an XML to UI dialog box using the XML definition in the skew.xml file
var skewlistDlg = fl.getDocumentDOM().xmlPanel( fl.configURI + "Commands/
  skewlist.xml" );

// Place the values of xskew and yskew from the dialog box into local
  variables.
// Note that we cast (convert) the return value of skewlistDlg["xSkew"] to a
  number before assigning
// it to xSkew because the skewSelection method takes numbers as parameters.
var xSkew = Number(skewlistDlg.xSkew);
var ySkew = Number(skewlistDlg.ySkew);
var edge = skewlistDlg.edge;

if (skewlistDlg.dismiss == "accept") {

  // Place the values of xSkew and ySkew from the dialog box
  // into local variables. The code casts (converts) the values from the
  // dialog box to a number before assigning them to the local variables
  // because the skewSelection() method takes numbers for
  // the xSkew and ySkew parameters.
  var xSkew = Number(skewlistDlg.xSkew);
  var ySkew = Number(skewlistDlg.ySkew);
  var edge    = skewlistDlg.edge;

  // check for valid input because sending 0 or undefined to
  // skewSelection() will cause the object to disappear.
  var inputIsValid = true;
  if (xSkew == 0 || isNaN(xSkew)) {
    inputIsValid = false;
  }
  if (ySkew == 0 || isNaN(ySkew)) {
    inputIsValid = false;
  }
```

```
  // Call skewSelection() to carry out the resizing command.
  if (inputIsValid ) {
    fl.getDocumentDOM().skewSelection(xSkew, ySkew, edge);
  }
}
```

Second, create a file named skewlist.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<dialog id="skewlist-dialog" title="Skew Selection" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="left">
        <label value="Skew x: " control="xSkew" align="left"/>
        <popupslider id="xSkew" minvalue="-180" maxvalue="180"/>
      </row>
      <row align="left">
        <label value="Skew y:" control="ySkew" align="left"/>
        <popupslider id="ySkew" minvalue="-180" maxvalue="180"/>
      </row>
      <row align="left">
        <label value="Edge:" control="edge" align="left"/>
        <listbox id="edge" rows="5">
          <listitem label="top center"/>
          <listitem label="right center"/>
          <listitem label="bottom center"/>
          <listitem label="left center"/>
        </listbox>
      </row>
    </rows>
  </grid>
</dialog>
```

The `skewlist` command now appears on the Commands menu. Draw a shape on the stage, and select it with the pointer tool. If you select the `skewlist` command from the Commands menu, the dialog box defined by skewlist.xml appears, as shown in the following figure:



<listitem>

Availability

Flash MX 2004.

Usage

```
<listitem
  label="mylabel"
  value="myValue"/>
```

Attributes

`label` String; text that appears in the listbox for that item.

`value` String; text that is returned if the user selects the item. If not set, the value of the `label` attribute is returned.

Child tags

None.

Parent tag

`<listbox>`

Description

Control tag; creates an individual item in a list box control. This tag must be used within a `<listbox>` tag.

Example

See the example for `<listbox>`.

<menulist>

Availability

Flash MX 2004.

Usage

```
<menulist
  id="myID"
  tabindex="myIdx">
  <menupop>
    <menuitem/>
    ...
    <menuitem/>
  </menupop>
</menulist>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`tabindex` Number; an integer used to set the control's position in the tab order (available only on Windows).

Child tags

`<menupop>`

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a pop-up menu control that contains `<menupop>` and `<menuitem>` tags.

Example

The following example uses the JavaScript API to create a new `Convert to Symbol` command that appears on the Commands menu. This command is a simple version of the Convert to Symbol dialog box that is on the Modify menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

First, create a file named Convert to Symbol.jsfl and place it in your Commands folder. Place the following code into the file and save the file:

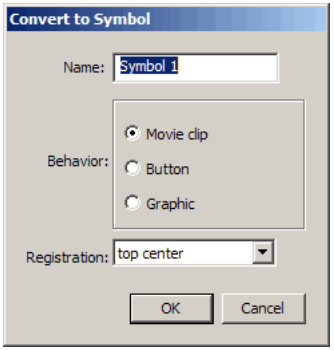
```
// Create an XML to UI dialog box using the XML definition in the
// Convert to Symbol.xml file
var convertToSymbolDlg = fl.getDocumentDOM().xmlPanel( fl.configURI +
  "Commands/Convert to Symbol.xml" );

if (convertToSymbolDlg.dismiss == "accept") {
  var type = new String(convertToSymbolDlg.type);
  fl.getDocumentDOM().convertToSymbol(type.toLowerCase(),
  convertToSymbolDlg.name, convertToSymbolDlg.registration);
}
```

Second, create a file named Convert to Symbol.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<dialog id="convertToSymbolDlg" title="Convert to Symbol" buttons="accept,
  cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="right">
        <label value="Name: " control="name" align="right"/>
        <textbox id="name" value="Symbol 1"/>
      </row>
      <row><spacer/></row>
      <row align="right">
        <label value="Behavior:" control="type" align="right"/>
        <radiogroup id="type">
          <radio label="Movie clip"/>
```

```
            <radio label="Button"/>
            <radio label="Graphic"/>
          </radiogroup>
        </row>
        <row align="right">
          <label value="Registration:" control="registration"/>
          <menulist id="registration">
            <menupop>
              <menuitem label="top left"/>
              <menuitem label="top center"/>
              <menuitem label="top right"/>
              <menuitem label="center left"/>
              <menuitem label="center"/>
              <menuitem label="center right"/>
              <menuitem label="bottom left"/>
              <menuitem label="bottom center"/>
              <menuitem label="bottom right"/>
            </menupop>
          </menulist>
        </row>
      </rows>
    </grid>
</dialog>
```

The `Convert to Symbol` command now appears on the Commands menu. Draw a shape on the stage, and select it with the pointer tool. If you then select the `Convert to Symbol` command from the Commands menu, the dialog box defined by Convert to Symbol.xml appears, as shown in the following figure:



<menupop>

Availability

Flash MX 2004.

Usage

```
<menulist>
  <menupop>
    <menuitem/>
    ...
```

```
      <menuitem/>
    </menupop>
  </menulist>
```

Attributes

None.

Child tags

<menuitem>

Parent tag

<menulist>

Description

Control tag; creates the pop-up menu of a pop-up menu control, and must contain at least one <menuitem> tag.

Example

The following example creates a pop-up menu control with eight elements. To see the XML definition of the entire dialog box, see the example for <menulist>.

```
<menulist id="registration">
  <menupop>
    <menuitem label="top left"/>
    <menuitem label="top center"/>
    <menuitem label="top right"/>
    <menuitem label="center left"/>
    <menuitem label="center"/>
    <menuitem label="center right"/>
    <menuitem label="bottom left"/>
    <menuitem label="bottom center"/>
    <menuitem label="bottom right"/>
  </menupop>
</menulist>
```

<menuitem>

Availability

Flash MX 2004.

Usage

```
<menulist>
  <menupop>
    <menuitem
      label="displayText"
      value="itemValue"/>
    ...
    <menuitem
      label="displayText"
      value="itemValue"/>
  </menupop>
```

Attributes

`label` String; text that appears in the pop-up menu for that item.

`value` String; text that is returned if the user selects the item. If not set, the value of the `label` attribute is returned.

Child tags

None.

Parent tag

`<menupop>`

Description

Control tag; creates the pop-up aspect of a pop-up menu control, and must contain at least one `<menuitem>` tag.

Example

The following example creates a drop-down menu with eight elements. To see the XML definition of the entire dialog box, see the example for `<menulist>`.

```
<menulist id="registration">
  <menupop>
    <menuitem label="top left"/>
    <menuitem label="top center"/>
    <menuitem label="top right"/>
    <menuitem label="center left"/>
    <menuitem label="center"/>
    <menuitem label="center right"/>
    <menuitem label="bottom left"/>
    <menuitem label="bottom center"/>
    <menuitem label="bottom right"/>
  </menupop>
</menulist>
```

<popupslider>

Availability

Flash MX 2004.

Usage

```
<popupslider
  id="myLabel"
  tabindex=""
  minvalue=""
  maxvalue=""/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`tabindex` Number; an integer that represents the control's position in the tab order (available only on Windows).

`minvalue` Number; an integer that represents the minimum value.

`maxvalue` Number; an integer that represents the maximum value.

Child tags

None.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a pop-up slider control (this is not part of the XUL standard).

Example

The following example uses the JavaScript API to create a new command that appears on the Commands menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

First, create a file named skew.jsfl and place it in your Commands folder. Place the following code into the file and save the file:

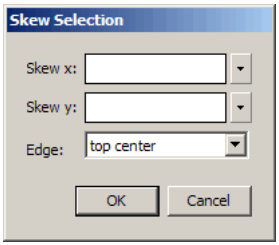
```
// Create an XML to UI dialog box using the XML definition in the skew.xml file
var skewDlg = fl.getDocumentDOM().xmlPanel( fl.configURI + "Commands/skew.xml"
  );

// Place the values of xskew and yskew from the dialog box into local
  variables.
// Note that we cast (convert) the return value of skewDlg["xSkew"] to a number
  before assigning
// it to xSkew because the skewSelection method takes numbers as parameters.
var xSkew = Number(skewDlg.xSkew);
var ySkew = Number(skewDlg.ySkew);
var edge = skewDlg.edge;

if (skewDlg.dismiss == "accept") {

  // Place the values of xSkew and ySkew from the dialog box
  // into local variables. The code casts (converts) the values from the
  // dialog box to a number before assigning them to the local variables
  // because the skewSelection() method takes numbers for
  // the xSkew and ySkew parameters.
  var xSkew = Number(skewDlg.xSkew);
  var ySkew = Number(skewDlg.ySkew);
  var edge    = skewDlg.edge;

  // check for valid input because sending 0 or undefined to
  // skewSelection() will cause the object to disappear.
  var inputIsValid = true;
  if (xSkew == 0 || isNaN(xSkew)) {
```

```
    inputIsValid = false;
  }
  if (ySkew == 0 || isNaN(ySkew)) {
    inputIsValid = false;
  }

  // Call skewSelection() to carry out the resizing command.
  if (inputIsValid ) {
    fl.getDocumentDOM().skewSelection(xSkew, ySkew, edge);
  }
}
```

Second, create a file named skew.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<dialog id="skew-dialog" title="Skew Selection" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="left">
        <label value="Skew x: " control="xSkew" align="left"/>
        <popupslider id="xSkew" minvalue="-180" maxvalue="180"/>
      </row>
      <row align="left">
        <label value="Skew y:" control="ySkew" align="left"/>
        <popupslider id="ySkew" minvalue="-180" maxvalue="180"/>
      </row>
      <row align="left">
        <label value="Edge:" control="edge" align="left"/>
        <menulist id="edge">
          <menupop>
            <menuitem label="top center"/>
            <menuitem label="right center"/>
            <menuitem label="bottom center"/>
            <menuitem label="left center"/>
          </menupop>
        </menulist>
      </row>
    </rows>
  </grid>
</dialog>
```

The skew command now appears on the Commands menu. Draw a shape on the stage, then select it with the pointer tool. If you then select the skew command from the Commands menu, the dialog box defined by skew.xml appears, as shown in the following figure:



<property>

Category

Flash MX 2004.

Usage

```
<property
  id="myID"/>
```

Attributes

id String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

Child tags

None.

Parent tag

<flash>

Description

Control tag; creates a custom property in an embedded SWF file; used with the <flash> tag. This tag is used to declare properties that are specific to a SWF file that is embedded in an XML to UI dialog box.

Example

See the example for <flash>.

<radiogroup>

Availability

Flash MX 2004.

Usage

```
<radiogroup
  id = "myID"
```

```
    label = "myLabel"
    groupbox = "true|false">
    <radio/>
    ...
    <radio/>
</radiogroup>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`label` String; defines a string that appears above the group of radio buttons.

`groupbox` Boolean value; if `true`, the entire group of radio buttons have a border around them.

Child tags

`<radio>`

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; Creates a container for a group of radio button controls. This tag allows grouping of radio buttons and must contain at least one `<radio>` tag.

Example

The following example defines a group of `<radio>` controls that are included in the example for `<menulist>`. For the complete example, see `<menulist>`.

```
<radiogroup id="type">
  <radio label="Movie clip"/>
  <radio label="Button"/>
  <radio label="Graphic"/>
</radiogroup>
```

<radio>

Availability

Flash MX 2004.

Usage

```
<radiogroup>
  <radio label="myLabel" selected="" accesskey=""/>
  ...
</radiogroup>
```

Attributes

`label` Text that appears next to the radio button.

`selected` Boolean value; if `true`, it makes the radio button the default selection in the radio group.

`accesskey` String; a character to be used for the keyboard shortcut for this control (available only on Windows).

Child tags

None.

Parent tag

`<radiogroup>`

Description

Control tag; creates a single radio button control. This tag must be used within a `<radiogroup>` tag.

Example

The following example defines a group of `<radio>` controls that are included in the example for `<menulist>`. For the complete example, see `<menulist>`.

```
<radiogroup id="type">
  <radio label="Movie clip"/>
  <radio label="Button"/>
  <radio label="Graphic"/>
</radiogroup>
```

<targetlist>

Availability

Flash MX 2004.

Usage

```
<targetlist
  id="myLabel"
  class="myClass1[, myClass2][, ..., myClassN]"
  required="true|false"
  pathtype="relative|absolute"/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`class` The class or classes for which to list instances.

`required` Boolean value; if `true`, the OK button cannot function until a value is entered for this control; if `false`, the control has no effect on the OK button.

`pathtype` String; the two possible values are `relative` and `absolute`.

Child tags

None.

Description

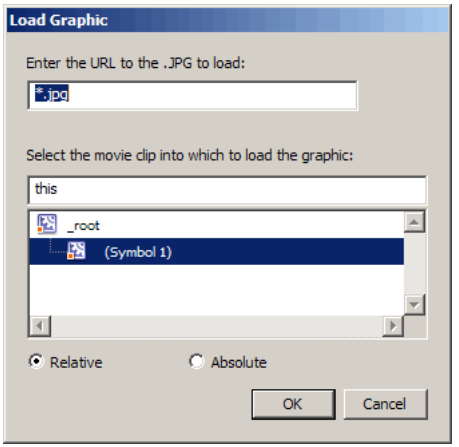
Control tag; creates a control that lists all instances of a class and lets the user select an instance.

Example

The following example is excepted from the Load Graphic behavior definition file. The tags define a targetlist control that lets users select a movie clip into which a graphic will be loaded. For the complete dialog box definition, see the Graphic_load_graphic.xml file in the Behaviors folder.

```
<vbox>
  <label value=""/>
  <label value="Select the movie clip into which to load the graphic:"/>
  <targetlist id="target" class="movieclip"/>
</vbox>
```

The control created by the `<targetlist>` tag is shown in the following figure:



<textbox>

Availability

Flash MX 2004.

Usage

```
<textbox
  id = "myID"
  literal = "true|false"
  maxlength = "myLength"
  multiline = "true|false"
  size = "mySize"
  tabindex = "myIdx"
  value = "myValue"/>
```

Attributes

`id` String; represents a unique identification string that is used by the extensibility features to identify the control and access the value it returns.

`literal` Boolean value; if `true`, the value that is returned from this control is enclosed by quotation marks (""). If `false`, which is the default setting, the returned value does not have quotation marks.("").

`maxlength` Number; sets the maximum number of characters that can be entered.

`multiline` Boolean value; if `true`, more than one line of input is allowed. If `false`, which is the default setting, only one line of input is allowed.

`size` Number; an integer that sets the width of the input field using the average character width.

`tabindex` Number; an integer that represents the control's position in the tab order (available only on Windows).

`value` String; default text that appears in the textbox.

Child tags

None.

Parent tag

`<dialog>`, `<hbox>`, `<row>`, `<vbox>`

Description

Control tag; creates a control that allows input of text.

Example

The following example uses the JavaScript API to create a new command that appears on the Commands menu. Create two files, as described in this section, and place them in your Commands folder in your user-level configuration folder. For more information, see "Configuration folders installed with Flash" in *Getting Started with Flash*.

First, create a file named scale.jsfl in your Commands folder. Place the following code into the file and save the file:

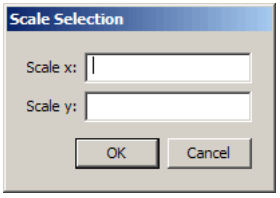
```
// Create an XML to UI dialog box using the XML definition
// in the scale.xml file
var scaleDlg = fl.getDocumentDOM().xmlPanel( fl.configURI + "Commands/
  scale.xml" );

if (scaleDlg.dismiss == "accept") {

  // Place the values of xScale and yScale from the dialog box
  // into local variables. The code casts (converts) the values from the
  // dialog box to a number before assigning them to the local variables
  // because the scaleSelection() method takes numbers as parameters.
  var xScale = Number(scaleDlg.xScale);
  var yScale = Number(scaleDlg.yScale);
```

```
  // check for valid input because sending 0 or undefined to
  // scaleSelection() will cause the object to disappear.
  var inputIsValid = true;
  if (xScale == 0 || isNaN(xScale)) {
    inputIsValid = false;
  }
  if (yScale == 0 || isNaN(yScale)) {
    inputIsValid = false;
  }

  // Call scaleSelection to carry out the resizing command.
  if (inputIsValid ) {
    fl.getDocumentDOM().scaleSelection(xScale, yScale);
  }
}
```

Second, create a file named scale.xml and place it in your Commands folder. Place the following code into the file and save the file:

```
<?xml version="1.0"?>
<dialog id="scale-dialog" title="Scale Selection" buttons="accept, cancel">
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>
    <rows>
      <row align="center">
        <label value="Scale x: " control="xScale"/>
        <textbox id="xScale"/>
      </row>
      <row align="center">
        <label value="Scale y:" control="yScale"/>
        <textbox id="yScale" />
      </row>
    </rows>
  </grid>
</dialog>
```

The `scale` command now appears on the Commands menu. Draw a shape on the Stage and select it with the pointer tool. If you select the `scale` command from the Commands menu, the dialog box defined by scale.xml appears, as shown in the following figure:



INDEX

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X