

ADOBE® FLASH® CS4 PROFESSIONAL

CLASSROOM IN A BOOK®

Instructor Notes



© 2009 Adobe Systems Incorporated and its licensors. All rights reserved.

Adobe® Flash CS4 Professional Classroom in a Book® for Windows® and Mac OS

If this guide is distributed with software that includes an end user agreement, this guide, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. Except as permitted by any such license, no part of this guide may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of Adobe Systems Incorporated. Please note that the content in this guide is protected under copyright law even if it is not distributed with software that includes an end user license agreement.

The content of this guide is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies that may appear in the informational content contained in this guide.

Please remember that existing artwork or images that you may want to include in your project may be protected under copyright law. The unauthorized incorporation of such material into your new work could be a violation of the rights of the copyright owner. Please be sure to obtain any permission required from the copyright owner.

Any references to company names in sample files are for demonstration purposes only and are not intended to refer to any actual organization.

Adobe, the Adobe logo, Acrobat, Classroom in a Book, Illustrator, InDesign, Photoshop, PostScript, and PostScript 3 are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Apple, Mac, and Macintosh are trademarks of Apple, registered in the U.S. and other countries. Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation registered in the U.S. and/or other countries. All other trademarks are the property of their respective owners.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110-2704, USA

Notice to U.S. Government End Users. The Software and Documentation are “Commercial Items,” as that term is defined at 48 C.F.R. §2.101, consisting of “Commercial Computer Software” and “Commercial Computer Software Documentation,” as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. Consistent with 48 C.F.R. §12.212 or 48 C.F.R. §§227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished-rights reserved under the copyright laws of the United States. Adobe Systems Incorporated, 345 Park Avenue, San Jose, CA 95110-2704, USA. For U.S. Government End Users, Adobe agrees to comply with all applicable equal opportunity laws including, if appropriate, the provisions of Executive Order 11246, as amended, Section 402 of the Vietnam Era Veterans Readjustment Assistance Act of 1974 (38 USC 4212), and Section 503 of the Rehabilitation Act of 1973, as amended, and the regulations at 41 CFR Parts 60-1 through 60-60, 60-250, and 60-741. The affirmative action clause and regulations contained in the preceding sentence shall be incorporated by reference.

Adobe Press books are published by Peachpit, Berkeley, CA. To report errors, please send a note to errata@peachpit.com.

Printed in the USA

Book:

ISBN-13: 978-0-321-57382-7

ISBN-10: 0-321-57382-X

Instructor notes:

ISBN-13: 978-0-321-61929-7

ISBN-10: 0-321-61929-3

INSTRUCTOR NOTES

Getting Started

The lessons in this book require assets, including Flash files that are included in the Lessons folder on the book's CD. Before beginning to work with the lessons, copy the files to each student's computer's hard disk.

Though Flash CS4 Professional requires a minimum of 512 MB RAM, Adobe strongly recommends using this book with a computer that has at least 1 GB RAM, especially when using larger video files, as in Lesson 7.

You may wish to use all the lessons in this book, or particular lessons. You can use the lessons in any order, but some tools and techniques are described in detail only in the first lesson in which they occur.

Introduce students to additional resources, such as Adobe Design Center and Adobe Developer Center, where they can access information provided by other Flash users and by Adobe.

Lesson 1: Getting Acquainted

Lesson 1 uses a sample project to provide an overview of the Flash workspace. Students are introduced to the Tools panel, Library panel, Property inspector, and Timeline. They also learn about layers and keyframes.

Starting Flash and opening a file

Demonstrate multiple ways of opening a document in Flash: double-clicking the Flash document, or by starting Flash and choosing File > Open, and selecting the document from the Welcome screen that appears when you start Flash. Explain the difference between the SWF file and the FLA file. The SWF file is the final, published Flash document, while the FLA is the editable source document.

Emphasize the importance of saving the working copy of the file to the lesson's Start folder. In many lessons, Flash documents reference other asset files, which must be in the same folder as the Flash document.

Getting to know the work area

Identify and describe the following areas of the workspace: Timeline, Tools panel, Stage, Library panel, Property inspector, and the Edit bar above the Stage. Explain to the students how they can undock and move the panels around to suit their working style, and how to return to the default layout by choosing Window > Workspace > Essentials.

If you have Adobe Creative Suite 4 installed, you can show students the similarities between panels in Flash CS4 and panels in other Creative Suite 4 applications. The Timeline is similar to the Timeline in Adobe Premiere Pro or Adobe After Effects. The Property inspector is similar to the one in Dreamweaver.

Understanding the Timeline

Layers overlap each other in the order in which they appear in the Timeline, with the top layer overlapping the bottom layers. As designers plan projects, they should consider the stacking order of the layers, and to be aware of how the objects in different layers appear in front of or behind objects in other layers.

Invite students to move the playhead through (or scrub) the Timeline to see the animation. As you move the playhead, Flash displays the contents of the frame on the Stage. Hide and show different layers on frames to demonstrate how layers are stacked and that each frame may contain content from multiple layers.

Keyframes are essential in animation. A keyframe indicates a change, whether it be in the position or other qualities of something on the Stage or the beginning or end of an audio file. Students will use keyframes in every Flash project, so it's important that they understand the keyframe's purpose clearly.

Describe the difference between inserting a keyframe and inserting a blank keyframe. When you insert a keyframe, the keyframe contains the frame's original contents until you change it. When you insert a blank keyframe, all content is removed from the frame so that you can add new content. Also, describe the difference between a frame and a keyframe. A frame is a measure of time on the Timeline. A keyframe is represented on the Timeline with a circle and indicates a change in content on the Stage.

Using the Property inspector

Demonstrate that the options in the Property inspector change to reflect what's currently selected. The Property inspector can display properties for the entire document, the selected frame, the selection on the Stage, or the selected tool in the Tools panel.

Using the Tools panel

The Tools panel contains several tools for selecting, drawing, editing, and navigating. Point out that the main tool for selecting and moving objects is the Selection tool. Demonstrate that some tools are grouped together, and a tool may be hidden beneath another. To select a hidden tool, click the triangle in the icon of the displayed tool, and then select the hidden tool from the pop-up menu.

Beneath the tools is the tools options area. The options available depend on the selected tool. Demonstrate that the options area changes as you select different tools. When you select the Rectangle tool, the Object Drawing mode icon appears; when you select the Zoom tool, the Enlarge and Reduce icons are available; when you select the Pencil tool, the Pencil mode icons are available. Show the students that additional options for a selected tool are also available in the Property inspector.

Lesson 2: Working with Graphics

Lesson 2 uses a sample project to introduce students to the drawing tools and to creating simple graphics. The project is a static banner ad of a glass of sparkling water over a wavy background and a text logo. As they create the banner, students learn how to use the various drawing, editing, and selection tools in the Tools panel. They learn to create basic shapes, import a bitmap fill, make selections, group objects, make patterns and align objects, create gradients and transparencies, create curves, and manipulate objects with the Free Transform tool. They also learn about the difference between drawing modes in Flash.

Flash drawing modes

By default, Flash uses the merge drawing mode when you draw rectangles, ovals, or lines. In merge drawing mode, the drawn elements are called shapes. Overlapping shapes merge together to form a single element on the Stage. When you select a shape, the selected area appears dotted. Though the elements merge together, you can select each separately. Each side of a rectangle's stroke is a separate element, for example. To select a shape's fill and stroke, drag the Selection tool around the entire shape. Demonstrate to the student how to select the fill or the stroke of a shape by clicking on it; or to select the entire shape (both the fill and the stroke) by double-clicking on it.

In object drawing mode, active when the Object Drawing icon is selected in the options area of the Tools panel, rectangles, ovals, and lines are discrete elements, even when they overlap. The discrete elements are called drawing objects. When you select a drawing object, its fill and stroke are both selected.

In primitive drawing mode, active when you use the Rectangle Primitive or Oval Primitive drawing tools, drawn elements are discrete objects, but you can modify their corner angles or inner radius.

Merge drawing mode offers advantages in creating unusual, complex shapes. Object drawing mode lets you move objects without concern that they will overlap and clip other objects. Primitive drawing mode makes it possible to draw more complex shapes without having to merge multiple shapes together.

Invite students to draw and manipulate elements using each of these drawing modes so that they become familiar with the differences and can recognize them as they work with them. You can convert a shape to an object by selecting it and choosing **Modify > Combine Object > Union**. You can convert an object to a shape by selecting it and choosing **Modify > Break Apart**.

Breaking apart and grouping objects

Emphasize for students the difference between shapes and groups in Flash, and the advantage of groups when you want to align certain elements of a drawing with the Align panel. Demonstrate how you can have groups within groups, and that you can always break apart groups to their component parts.

Creating transparencies

Transparency, or inversely, opacity, is determined by the alpha value of a color. Decreasing the alpha value decreases the opacity and increases the transparency. Alpha values range from 0 (totally transparent) to 100 (totally opaque). Students can change the alpha value for a color in the Color panel or from the Alpha field in the Fill options of the Tool panel.

Creating and editing text

Creating text in Flash is similar to creating text in Adobe InDesign, Adobe Photoshop, or numerous other applications. Use the Text tool to create an insertion point on the Stage and begin typing, or drag the Text tool to create a text box for your text. In this lesson, students use Static text, which is used for text that does not change and cannot be controlled with ActionScript. To format text, select it with the Text tool, and then select formatting options in the Property inspector. To manipulate a text box as an object, select it with the Selection tool and make changes in the Property inspector or Transform panel, or simply drag the text box handles to resize it. Resizing the text box does not resize the text itself, but it does reflow the text in the text box. You can create a hyperlink from any part of the text, and you can also apply a filter, such as a drop shadow or glow, to text.

Lesson 3: Creating and Editing Symbols

Lesson 3 uses a sample project to introduce symbols, instances, and imported Illustrator and Photoshop files as students create a static frame of a cartoon. Students learn to create new symbols and to convert selections on the Stage to symbols. Students learn how to edit the symbols from the Library panel and to edit the symbols in place on the Stage. Students also learn to adjust the color effect and blending, apply filters, and move the symbol instances in three-dimensional space.

Importing Illustrator files

Flash lets you import Illustrator AI files, and to a large extent preserves the artwork's editability and visual fidelity. The AI Importer also provides a great degree of control in determining how Illustrator artwork is imported into Flash, letting you specify how to import specific objects into an AI file.

The Flash AI Importer provides the following key features:

- Preserves editability of the most commonly used Illustrator effects such as Flash filters.
- Preserves editability of blend modes that Flash and Illustrator have in common.
- Preserves the fidelity and editability of gradient fills.
- Maintains the appearance of RGB (red, green, blue) colors.
- Imports Illustrator Symbols as Flash Symbols.
- Preserves the number and position of Bezier control points.
- Preserves the fidelity of clip masks.
- Preserves the fidelity of pattern strokes and fills.
- Preserves object transparency.
- Converts the AI file layers to individual Flash layers, keyframes, or a single Flash layer. You can also import the AI file as a single bitmap image, in which case Flash flattens (rasterizes) the file.
- Provides an improved copy-and-paste workflow between Illustrator and Flash. A copy-and-paste dialog box provides settings to apply to AI files being pasted onto the Flash stage.

You specify how artwork is imported in the Import dialog box. You can select which layers—and which content on each layer—to import. Flash can import Illustrator layers as Flash layers, as a single layer, as keyframes, or as symbols.

In this lesson, students import an Illustrator file with the characters of the cartoon frame. Many of the same options are available when copying and pasting artwork. If Illustrator is installed on the computer, open the Illustrator file and copy and paste, or drag and drop, artwork from it to the Stage in Flash to demonstrate how the options are similar.

About symbols

A *symbol* is a reusable asset that you can use for special effects, animation, or interactivity. There are three kinds of symbols: the graphic, button, and movie clip. Symbols can reduce the file size and download time for many animations because they can be reused. You can use a symbol countless times in a project, but Flash includes its data only once.

Symbols are stored in the Library panel. When you drag a symbol to the Stage, Flash creates an *instance* of the symbol, leaving the original in the library. An instance is a copy of a symbol located on the Stage. You can offer your students the analogy of a symbol as a photographic negative. The instances on the Stage can be thought of as prints of the negative. With just a single negative, you can create multiple prints.

Movie clip symbols, button symbols, and graphic symbols each have unique functions, benefits, and limitations. Create one of each, and then show students the differences in the Timeline and in the Property inspector for each.

Creating symbols

In Flash, there are two ways to create a symbol. The first is to have nothing on the Stage selected, and then choose Insert > New Symbol. Flash brings you to symbol-editing mode, where you can begin drawing or importing graphics for your symbol.

The second way is to select existing graphics on the Stage, and then choose Modify > Convert to Symbol (F8). Whatever is selected will automatically be placed inside your new symbol. The “Convert to Symbol” language may confuse students, as there is really no “conversion”. Explain that the selected items are simply placed inside of the newly created symbol.

The registration point of a symbol is indicated by the crosshair on the Stage in symbol-editing mode. The registration point determines the point at which Flash reports the coordinates of the symbol, and from which it resizes or transforms the symbol.

Editing and managing symbols

Demonstrate to the students the two ways in which you can edit your symbols. First, you can double-click the symbol in the Library and edit its contents in symbol-editing mode. Return to the main Timeline by clicking the Scene 1 button on the top horizontal bar above the Stage. Secondly, you can double-click the instance on the Stage and edit the symbol in place. Editing in place lets you see all the other elements surrounding the instance on the Stage. When you edit a symbol, all of its instances on the Stage change to reflect the new edits.

Changing the appearance of instances

There are a number of ways you can change the appearance of instances. You can change the size of any instance with the Free Transform tool. You can also change the brightness, tint, or transparency of an instance by setting different values for the instance's Color Effect in the Property inspector. You can also apply a Blend effect or a Filter on an instance. Blending refers to how the colors of an instance interact with the colors below it. Invite the students to explore the different Blend options. Filters are special effects, such as blurs and glows, which you can apply to instances. Finally, new to Flash CS4, you can position or rotate any instance in three-dimensional space using the 3D Translation and 3D Rotation tool.

Lesson 4: Adding Animation

Lesson 4 uses an animated motion picture promotional website to introduce students to motion tweens, easing, editing the paths of motion, the Motion Editor, and nested animations.

About animation

Animation is the movement, or change, of objects through time. Animation can be as simple as moving a box across the Stage from one frame to the next. It can also be much more complex. You can animate many different aspects of a single object: the position, the color or transparency, the size or rotation, and filters. You can also animate an object's position and rotation in three-dimensional space. You can control their path of motion, and their easing, which is the way an object accelerates or decelerates.

The basic workflow for animation goes like this: To animate objects in Flash, you select the object on the Stage, right-click/Ctrl-click, and choose Create Motion Tween from the context menu. Move the playhead to a different point in time and move the object to a new position.

Motion tweens require you to use a symbol instance. If the object you've selected is not a symbol instance, Flash will automatically ask to convert the selection to a symbol. Flash also automatically separates motion tweens on their own layers, which are called Tween layers. There can only be one motion tween per layer without any other element in the layer. Tween layers allow you to change various attributes of your instance at different keypoints over time.

Changing the pacing and timing

Simply click and drag the end of a tween span to add more frames to it and expand the amount of time it takes to do the animation. All the keyframes within the tween span will be distributed evenly. You can also shift-drag the end of the tween span to add frames without affecting the timing of the keyframes. Remind students that they can also right-click/Ctrl-click on the Timeline and choose Insert Frames to add frames, or press F5 on the keyboard to add frames.

To move individual keyframes, first Control-click (Windows) or Command-click (Mac) to select the keyframe, and then click and drag it to a new position.

Demonstrate to the students how the positions of the keyframes affect the timing of the animation, and how the total length of the tween span affects the overall speed of the animation.

Changing the path of the motion

The path that an object takes as it moves on the Stage during a motion tween is the motion path, and it is represented by a colored line. You can edit the path of the motion with the Free Transform tool, or by using the Selection tool, or, if you want Bezier precision, you can use the Convert Anchor Point tool and Subselection tool.

Creating nested animations

Movie clip symbols have their own Timelines, similar to the main Timeline. A movie clip symbol may have other symbols nested within it. In this lesson, the students create a nested animation of the alien waving his arms and another nested animation of the car jerking up and down.

Using the Motion Editor

The Motion Editor is a panel that provides in-depth information and editing capabilities for all the properties of a motion tween. The Motion Editor is located behind the Timeline and can be accessed by clicking on the top tab or by choosing Window > Motion Editor.

The Motion Editor shows how individual properties change over time as a graph. You can add keyframes to a graph by clicking on the diamond icon or right-clicking/Ctrl-clicking on the graph and choosing Insert Keyframe. Drag the keyframe up or down to its new value, or enter the new amount in the Value column.

You can use the Motion Editor to apply sophisticated easing effects to any property. Easing refers to the way in which a motion tween proceeds. In the most basic sense, it can be thought of as acceleration or deceleration. Right-click/Ctrl-click a keyframe in the Motion Editor and select Smooth point to make handles appear from the keyframe. Drag the handles to change the curvature of the graph, which affects the easing for that property.

Lesson 5: Articulated Motion and Morphing

Lesson 5 introduces methods for animating two specialized kinds of motion: articulated motion and morphing. Articulated motion involves motion of an object that is made up of multiple, linked objects. Flash can animate a linked object (an *armature*) using inverse kinematics, a mathematical way to calculate how to move an armature from one pose to another. Morphing is the animation of shapes, and can create more organic changes in contours. Students explore both kinds of specialized animations with an animated crane and the ocean in one file, and an octopus in another file.

Articulated motion with inverse kinematics

The first step to create articulated motion is to define the bones of an object. You use the Bone tool to do that. The Bone tool tells Flash how a series of movie clip instances are connected. The connected movie clips are called the *armature*, and each movie clip is called a *node*. Armatures are separated on a Pose layer. Students learn to pose the crane at one point in time, and then create a new pose at a later point in time, and watch how Flash automatically animates the motion from one pose to the next.

Demonstrate how you can isolate the movement of each node by holding down the shift key as you move it. You can also edit the positions of the nodes by holding down the Alt (Windows) or Option (Mac) key and dragging them to new positions.

Constraining joints

The students are asked to constrain the various joints of the crane to explore the possibilities of the option. Many armatures in real life are constrained to certain angles of rotation. For example, your forearm can rotate up toward your bicep, but it can't rotate in the other direction beyond your bicep. When working with armatures in Flash CS4, you can choose to constrain the rotation for various joints or even constrain the translation (movement) of the various joints. Click on a bone in an armature and select Constrain in the Property inspector. Set the minimum and maximum allowable angles to constrain the joint.

Inverse kinematics with shapes

You can also create armatures for shapes, which are useful for animating objects without obvious joints and segments but can still have an articulated motion. In this part of the lesson, the students add bones to the arms of an octopus, in order to animate its undulating motion. To create an armature for a shape, select the Bone tool. Click and drag inside the selected shape to define the first bone and the subsequent connected bones.

The Bind tool can refine the connections between the shape and its armature. The Bind tool is hidden under the Bone tool. The Bind tool displays which control points are connected to which bones and lets you break those connections and make new ones.

Choose the Bind tool and click on any bone in the shape. The selected bone is highlighted in red, and all the connected control points on the shape are highlighted in yellow. If you want to redefine which control points are connected to the selected bone, you can do the following:

- Shift-click to add additional control points.
- Ctrl-click/Command-click to remove control points.
- Drag a connection line between the bone and the control point.

Click on any control point on the shape. The selected control point is highlighted in red, and all the connected bones are highlighted in yellow. If you want to redefine which bones are connected to the selected control point, you can do the following:

- Shift-click to add additional bones.
- Ctrl-click/Command-click to remove bones.
- Drag a connection line between the control point and the bone. In the following figure, another control point farther down the tentacle is being associated with the first bone.

Morphing with shape tweens

Shape tweening is a technique for interpolating amorphous changes between shapes in different keyframes. Shape tweens make it possible to smoothly morph one thing into another. Any kind of animation that requires that the contours of a shape change—for example, animation of clouds, water, or fire—is a perfect candidate for shape tweening. Students in this part of the lesson create two keyframes, each containing different shapes of the surface of the ocean. Applying a shape tween in between the two keyframes creates a smooth animation that morphs the shapes.

Emphasize to the students that both the fill and the stroke of a shape can be smoothly animated. Also point out that because shape tweening only applies to shapes, you can't use groups, symbol instances, or bitmap images.

Using shape hints

Shape hints force Flash to map points on the first shape to corresponding points on the second shape. By placing multiple shape hints, you can control more precisely how a shape tween appears. Insert a shape hint by selecting the first keyframe of a shape tween. Then, choose **Modify > Shape > Add Shape Hint**. A tiny, circled letter appears in the first keyframe and a corresponding circled letter appears in the next keyframe. Move both circled letters to matching points on the contours of the shapes.

A maximum of 26 shape hints are allowable, and for best results, place shape hints in a clockwise or counterclockwise fashion.

Lesson 6: Creating Interactive Navigation

Lesson 6 introduces ActionScript and the button symbol. Students learn how to structure the Timeline to create a project with non-linear navigation—a project in which the user chooses what content to see, and when to see it. The students learn about basic event handling and the commands for controlling the playhead. The end result is a photographer’s portfolio in which the user can click on any thumbnail image to see a larger version.

About interactive movies

Students should learn what it means to create non-linear navigation. In this lesson, students create a Flash movie that doesn’t play straight from the beginning to the end. ActionScript provides the instructions to make the Flash playhead to jump around and to go to different frames of the Timeline based on which button the user clicks. Different frames on the Timeline contain different content. The user doesn’t actually know that the playhead is jumping around the Timeline—all they see (or hear) is different content appearing as they click the buttons on the Stage.

Creating buttons

The buttons in this lesson include small thumbnail bitmap images. Review the three kinds of symbols and the benefits of each. Button symbols contain four unique keyframes. Demonstrate and explain each keyframe: Up, Over, Down, and Hit.

- The Up state determines the button’s default appearance when the mouse is not interactive with it.
- The Over state appears when the mouse rolls over the button.
- The Down state appears when the mouse is depressed over the button.
- The Hit state defines the button’s active area, which is the area that responds to the mouse rolling over or clicking the button.

A particularly effective demonstration of the Hit state is to delete half of the Hit state graphic in the button symbol. Test the movie with the button instance on the Stage and show how only the remaining half of the button is responsive to the mouse.

For more advanced students, show how button symbols with only their Hit states defined can be used as invisible buttons. Place an invisible button instance on the Stage to make that particular area clickable and responsive to the mouse.

Naming the button instances

Giving names to instances is a critical step in creating interactive Flash projects. Emphasize the importance of naming instances so ActionScript can refer to them. The most common beginner mistake is not to name, or to incorrectly name, a button instance. The instance names are not the same as the symbol names in the library. The names in the library are simple organizational reminders.

Instance naming follows these simple rules:

- Do not use spaces or special punctuation. Underscores are okay to use.
- Do not begin your name with a number.
- End your instance name with `_btn`. Although not required, it helps Flash identify those objects as buttons.

Name instances in a way that is meaningful, so you'll remember what you've called them when you're writing ActionScript. Shorter names are easier to type without error, but longer names may be easier to remember and to identify when you're editing the ActionScript later.

Understanding ActionScript 3.0

Adobe Flash CS4 uses ActionScript 3.0, the scripting language that extends Flash functionality and enables interactivity in Flash projects. Students who haven't worked with scripting languages before may initially be intimidated and confused by ActionScript, which is an object-oriented scripting language similar to JavaScript. This lesson introduces basic scripting concepts, terminology, and syntax, as well as specific ActionScript coding to make their buttons functional.

About ActionScript

The ActionScript scripting language lets you add complex interactivity, playback control, and data display to your application. You can add ActionScript in the authoring environment by using the Actions panel.

ActionScript follows its own rules of syntax, reserved keywords, and lets you use variables to store and retrieve information. ActionScript includes a large library of built-in classes that let you create objects to perform many useful tasks. For detailed information on ActionScript, see *Programming ActionScript 3.0* or the *ActionScript Language References*, both available in Flash Help.

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.

Because there are multiple versions of ActionScript, and multiple ways of incorporating it into your Flash document files, there are several different ways to learn ActionScript.

Flash Help describes the graphical user interface for working with ActionScript. This interface includes the Actions panel, Script window, Script Assist mode, Behaviors panel, Output panel, and Compiler Errors panel. These topics apply to all versions of ActionScript.

Other ActionScript documentation available in Flash Help will help you learn about the individual versions of ActionScript; see *Programming ActionScript 3.0*, *Learning ActionScript 2.0 in Adobe Flash*, *Developing Flash Lite 1.x Applications* or *Developing Flash Lite 2.x Applications*. For information about the ActionScript vocabulary, see the *ActionScript Language Reference* for the version you are working with.

For video tutorials about ActionScript 3.0, the Flash workflow, and components, see the following:

- Getting started with ActionScript 3.0: www.adobe.com/go/vid0129
- Creating interactivity with ActionScript 3.0: www.adobe.com/go/vid0130

Adding a stop action

Students have had some experience with ActionScript and the Actions panel in earlier lessons because they've added stop actions. Stop actions are very simple; they stop the movement in the Timeline and prevent files from looping.

As you discuss stop actions, talk about the role of stop actions added in earlier lessons. Remove a stop action from the end file of an earlier lesson and preview the movie. Discuss what happens when the stop file is no longer there. In most cases, the movie or some part of the movie repeats.

Discourage the use of Script Assist to add ActionScript to the Actions panel. Although it is meant to help users unfamiliar with coding, it is far more useful to teach students how to write code in the Script pane unassisted, rather than hunt for the code element in the Actions Toolbox and fill out the supplied fields.

Creating event handlers for buttons

Events are things that happen in the Flash environment that Flash can detect and respond to. Explain the concept of an event to the students, and give examples of various types of events. For example, a mouse click, a mouse movement, and a key press on the keyboard are all events. These events are produced by the user, but some events can happen independently of the user, like the successful loading of a piece of data, or the completion of a sound. With ActionScript, you can write code that detects events and respond to them with something called an event handler.

The first step in event handling is to create a listener that will detect the event. A listener looks something like this:

```
wheretolisten.addEventListener(whatevent, responsetoevent);
```

The actual command is `addEventListener()`. Clarify to the students which of the words are placeholders for objects and parameters for their situation. `Wheretolisten` is the object where the event occurs (usually a button), `whatevent` is the specific kind of event (such as a mouse click), and `responsetoevent` is a function that gets triggered when the event happens.

The next step is to create the function that will respond to the event. A function simply groups a bunch of actions together which you can trigger by referencing its name. A function looks something like this:

```
function responsetoevent (myEvent:MouseEvent){ };
```

In this particular example, the name of the function is called `responsetoevent`. It receives one parameter (within the parentheses), called `myEvent`, which is an event that involves a mouse event. If this function is triggered, all the actions between the curly braces are executed.

You should have the students memorize the most common mouse events and ActionScript commands for navigating the Timeline. The mouse events they should remember are:

- `MouseEvent.CLICK`
- `MouseEvent.DOUBLE_CLICK`
- `MouseEvent.MOUSE_MOVE`
- `MouseEvent.MOUSE_DOWN`
- `MouseEvent.MOUSE_UP`
- `MouseEvent.MOUSE_OVER`
- `MouseEvent.MOUSE_OUT`

The commands to control the playhead that they should remember are:

- `stop();`
- `play();`
- `gotoAndStop(framenumber or "frameLabel");`
- `gotoAndPlay(framenumber or "frameLabel");`
- `nextFrame();`
- `prevFrame();`

Creating destination keyframes

When the user clicks each button, Flash will move the playhead to a new spot on the Timeline. Students are asked to create new content at those particular frames. In this lesson, students learn to put new content in different keyframes, and to label each keyframe to make referencing them from ActionScript easier.

A few essential points for students to remember: Use double quotes around the name of the frame label in their ActionScript code. If they want to use the `gotoAndPlay()` command, then they will also need to add a `stop()` command on the Timeline to stop the playhead before it displays all the other keyframes along the Timeline.

Lesson 7: Working with Sound and Video

In Lesson 7, students bring audio and video files together to create a dynamic, interactive wildlife preserve kiosk. As they work through the project, students learn how to edit audio files, use the Adobe Media Encoder to encode their video files to the correct format, and to integrate video into their Flash project.

Using sounds

To add an audio file to a frame in Flash, first import it to the library. Then, select the frame at which you want the audio file to begin playing, and, in the Property inspector, choose the audio file from the Sound menu. You can also simply drag the sound file from the library to the Stage.

Selecting the appropriate option from the Sync pop-up menu is important. If a sound file doesn't play when you think it should, check the Sync menu to ensure you've selected the right option.

- **Event** synchronizes the sound to the occurrence of an event, such as a button click. An event sound plays when its first keyframe appears and plays in its entirety, even if the SWF file stops playing. If another event occurs, the first instance of the sound continues playing while the second instance of the sound plays.
- **Start** synchronizes the sound to the occurrence of an event, just as the Event option. However, if the sound is already playing, no new instance of the sound plays.
- **Stop** silences the specified sound.

● **Note:** Repeating stream sounds is not recommended, because frames are added to the file and the file size is increased by the number of times the sound is repeated. You can also apply effects to the sound file.

- **Stream** synchronizes the sound for playing on a website. Flash forces animation to keep pace with stream sounds. If it can't draw animation frames quickly enough to keep up with the sound file, it skips frames. Stream sounds stop if the SWF file stops playing. Next to the Sync menu is a pop-up menu that determines how many times the sound file plays.
- **Repeat** specifies the number of times the file plays; specify a number.
- **Loop** repeats the sound continuously.
- **None** applies no effect
- **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 Edit Envelope

Editing a sound clip

In Flash, you can change the point at which a sound starts and stops playing, or control the volume of the sound as it plays. In this lesson, students clip the end of the waveform of a sound to shorten the sound clip and edit the volume levels.

To edit a sound file, first add it to a frame, and click Edit in the Property inspector.

- 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.

Understanding Flash video

Before you import video into Flash, consider what video quality you need, what video format to use, and how it will be integrated with the rest of your Flash project. The two ways in which you can use video in Flash is: (1) to embed the video, or (2) to play the video from an external file. Embedding video requires that the video be in FLV format; playing from an external file requires that the video be in FLV or F4V format.

When you embed a video, it increases the size of the final SWF file that you publish. The embedded video appears on the Timeline and can be synchronized with other Flash elements.

When you play external video, the final SWF remains small because the video is kept separate from the Flash document. Playback of external video depends on a video playback component, which you can configure, and an optional skin, which provides the look-and-feel and functionality of the playback controls.

Tips for creating video for Flash with best results

Follow these guidelines to deliver the best possible video for Flash:

Work with video in the native format of your project until your final output.

If you convert a precompressed digital video format into another format such as FLV, the previous encoder can introduce video noise. The first compressor already applied its encoding algorithm to the video, reducing its quality, frame size, and rate. That compression may have also introduced digital artifacts or noise. This additional noise affects the final encoding process, and a higher data rate may be required to encode a good-quality file.

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 (as opposed to dissolves) are usually best. Eye-catching video sequences—for instance showing an object zooming from behind the first track, doing a “page peel,” or wrapping around a ball and then flying off the screen—don't compress well and should be used sparingly.

Know your audience data rate.

When you deliver video over the Internet, produce files at lower data rates. Users with fast Internet connections can view the files with little or no delay for loading, but dial-up users must wait for files to download. Make the clips short to keep the download times within acceptable limits for dial-up users.

● **Note:** When you embed video clips in the SWF file, the frame rate of the video clip must be the same as the frame rate of the SWF file.

Select the proper frame rate.

Frame rate indicates frames per second (fps). If you have a higher data rate clip, a lower frame rate can improve playback through limited bandwidth. For example, if you are compressing a clip with little motion, cutting the frame rate in half probably saves 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, leave the frame rate high if your delivery channels and playback platforms allow. For web delivery, get this detail from your hosting service. For mobile devices, use the device-specific encoding presets, and Device Central. 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 and frame aspect ratio.

At a given data rate (connection speed), increasing the frame size decreases video quality. When you select the frame size for your encoding settings, consider frame rate, source material, and personal preferences. To prevent pillarboxing, it's important to choose a frame size of the same aspect ratio as that of your source footage. For example, you get pillarboxing if you encode NTSC footage to a PAL frame size.

Know download times.

Know how long it will take to download enough of your video so that it can play to the end without pausing to finish downloading. While the first part of your video clip downloads, you may want to display other content that disguises the download. For short clips, use the following formula: $\text{Pause} = \text{download time} - \text{play time} + 10\% \text{ of play time}$. For example, if your clip is 30 seconds long and it takes one minute to download, give your clip a 33-second buffer ($60 \text{ seconds} - 30 \text{ seconds} + 3 \text{ seconds} = 33 \text{ seconds}$).

Remove noise and interlacing.

For the best encoding, you might need to remove noise and interlacing. 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 are barely noticeable on TV can be obvious on a computer screen.

Adobe Flash is intended for progressive display on computer screens and other devices, rather than on interlaced displays such as TVs. Interlaced footage viewed on a progressive display can exhibit alternating vertical lines in high-motion areas. Thus, all the Adobe Flash Video presets in the Adobe Media Encoder have deinterlacing turned on by default.

Follow the same guidelines for audio.

The same considerations apply to audio production as to video production. To achieve good audio compression, 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. To record from an analog source, use the highest-quality sound card available.

Using the Adobe Media Encoder

You can convert your video files into the FLV or F4V format using the Adobe Media Encoder CS4, a stand-alone application that comes with Flash CS4. The Adobe Media Encoder can convert single files or multiple files (known as batch-processing) to make your work flow easier.

You can customize many settings in the conversion of your original video to the Flash Video format. You can crop and resize your video to specific dimensions, just convert a snippet of the video, adjust the type of compression and the compression levels, or even apply filters to the video. To display the encoding options, click on the Preset selection in the display window, or choose Edit > Export Settings. The Export Settings dialog box appears.

Playback of external video

To play back an external FLV or F4V file, choose File > Import > Import Video. The Import Video wizard appears, which guides you through the process. During the process, you choose the external file that you want to play and an optional skin, which provides the look-and-feel and functionality of the controls.

Using external FLV files provides the following capabilities that are not available when using embedded video:

- You can use longer video clips 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.

Embedding Flash video

Embedded video lets you embed a video file within a SWF file. When you import video in this way, the video is placed in the Timeline where you can see the individual video frames represented in the Timeline frames. An embedded video file becomes part of the Flash document.

When you create a SWF file with embedded video, the frame rate of the video clip and the SWF file must be the same. If you use different frame rates for the SWF file and the embedded video clip, playback is inconsistent.

Embedded video works best for smaller video clips, with a playback time of less than 120 seconds. If you are using video clips with longer playback times, consider keeping and playing the video as an external file.

The limitations of embedded video include:

- You might encounter problems if the resulting SWF files become excessively large. Flash Player reserves a lot of memory when downloading and attempting to play large SWF files with embedded video, which can cause Flash Player to fail.
- Longer video files (over 120 seconds long) often have synchronization issues between the video and audio portions of a video clip. Over time, the audio track begins playing out of sequence with the video, causing a less than desirable viewing experience.
- There is a maximum length of 16,000 frames for embedded movies.

Lesson 8: Using Components

Lesson 8 introduces students to components, which are pre-built movie clip symbols that you can customize for interactive user interface elements. In the lesson, students create two Text Area components, one with ActionScript and one without, and a Tile List component.

About components

You can create components in Flash by dragging them from the Components panel or by creating them in ActionScript. To add components using the Flash interface, drag them from the Components panel onto the Stage, set their parameters in the Component inspector, and then assign additional behaviors and methods using ActionScript.

When you drag a component from the Components panel to the Stage, Flash imports an editable movie clip to the library. After a component has been imported to the library, you can drag instances of it to the Stage from either the Library panel or the Components panel.

You can set properties for each instance of a component in the Component inspector. You can also change the color and text formatting of a component by setting style properties for it or customize its appearance by editing the component's skins. Name component instances to refer to them in ActionScript.

Students can learn more about components by creating their own projects using components from the Components panel. Examples of components are included in the sidebar at the end of the lesson.

● **Note:** You can use the Behaviors panel to assign behaviors to components if the document's ActionScript Publish setting is set to ActionScript 2.0. In this book, we always use ActionScript 3.0, so behaviors in the Behaviors panel are not available.

Lesson 9: Loading and Controlling Flash Content

In this lesson, students learn how to keep a Flash project modular by loading external SWF files into a main SWF. In addition, they learn how to control a movie clip's Timeline and learn to create masks. The lesson is based on a fictional lifestyle magazine. Each section of the magazine is a separate SWF file that loads into the main SWF.

Loading external content

Loading external content keeps your overall project in separate modules and prevents the project from becoming too bloated and difficult to download. It also makes it easier for you to edit, because you can edit individual sections instead of one, large, unwieldy file.

Loading a SWF involves ActionScript. First, a Loader object must be created, as in the following statement:

```
var myLoader:Loader=new Loader();
```

Next, the load() method should be called for the Loader object. The parameter for the load() method is the URLRequest object, which defines the external SWF file.

```
var myURL:URLRequest=new URLRequest("page1.swf");  
myLoader.load(myURL);
```

Make sure to point out to the students that the filename for the SWF file must be enclosed in double quotation marks.

Finally, in order to display the loaded SWF on the main Stage, you need to add the Loader object to the display list with the following command:

```
addChild(myLoader);
```

In this lesson, students enter the code inside four separate event handlers.

Controlling movie clips

The initial animations are nested inside individual movie clips, and the students learn to control each of those movie clips' Timelines. After any one of the external SWFs is removed from the Stage, the playhead moves to the first frame of each of the movie clips and plays. Students learn to use the basic navigation commands that they learned in Lesson 6 (`gotoAndStop`, `gotoAndPlay`, `stop`, `play`) to navigate the Timelines of movie clips.

To navigate the Timeline of a movie clip, simply precede the navigation command with the name of the movie clip and separate them with a dot. Flash targets that particular movie clip and moves its Timeline accordingly.

Creating masks

Masking is a way of selectively hiding and displaying content on a layer. Masking is a way for you to control the content that your audience sees. Provide several examples of the kinds of effects that masking can do: for example, you can make a circular mask to create a spotlight effect.

In Flash, you put a mask on one layer and the content that is masked in a layer below it. After the students grasp the basic concept of masks, reveal the possibilities of an animation in the Mask layer or an animation in the Masked layer.

To create a mask, double-click the Layer icon in front of the layer name. In the Layer Properties dialog box that appears, select Mask. That layer becomes a Mask layer. Anything that is drawn in this layer will act as a mask for a masked layer below it.

To create the Masked layer, simply drag another normal layer under the Mask layer. Or, double-click the Layer icon in front of the layer name in the layer below the Mask layer. In the Layer Properties dialog box that appears, select Masked.

Warn students of the limitations of masking: Flash does not recognize different Alpha levels of a mask. For example, a mask at an Alpha value of 50% will still mask at 100%. However, with ActionScript you can dynamically create masks that will allow transparencies. Masks also do not recognize strokes.

Lesson 10: Publishing Flash Documents

In Lesson 10, students explore a range of options for publishing a document, using an alien animation that is provided for them.

Testing a Flash document

One fast way to preview a movie is to choose Control > Test Movie (Ctrl-Enter/ Cmd-Return). This command creates a SWF file in the same location as your FLA file so that you can play and preview the movie; it does not create the HTML file necessary to play the movie from a Web site. Remind students that the default behavior for your movie in the Test Movie mode is to loop. You can make your SWF play differently in a browser by selecting different publish settings, as described later in this chapter, or by adding ActionScript to stop the Timeline. Another option is to choose File > Publish Preview > Default (HTML) to export a SWF file and an HTML file required to play in a browser and to preview the movie.

Understanding the bandwidth profiler

You can preview how your final project might behave under different download environments by using the Bandwidth Profiler, a useful panel that is available when you are in Test Movie mode.

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.

To simulate download performance:

- 1 Choose Control > Test Movie.
- 2 In the preview window, choose View > Download Settings.
- 3 Select a download speed to determine the streaming rate that Flash simulates. When viewing the SWF file, choose View > Bandwidth Profiler to see 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 Choose View > Simulate Download to turn streaming off or on. If you turn streaming on, the document starts over and simulates a web connection.

Publishing a movie for the web

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

You can publish the Flash document 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 in alternative file formats, the settings for each file format are stored with the document.

You can export the Flash document in several formats, similar to publishing FLA files in alternative file formats, except that the settings for each file format are not stored with the FLA file.

Alternatively, create a custom HTML document with any HTML editor, such as Dreamweaver, and include the tags required to display a SWF file.

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. Before exporting a document, you can optimize it further by using various strategies to reduce the file size. Students should be made aware of the importance of file optimizations, as even the most gorgeous and astounding animations will never be seen if it is too large and cumbersome to download quickly. As you make changes, test your document by running it on a variety of computers, operating systems, and Internet connections.

- Use symbols, animated or otherwise, for every element that appears more than once.
- Use tweened animations whenever possible when creating animation sequences. Tweened animations use less file space than a series of keyframes.
- 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.

- Use mp3, the smallest sound format, whenever possible.
- Group elements.
- Use layers to separate elements that change during the animation from elements that do not.
- Use Modify > Shape > 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.
- 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.
- Use the Color menu in the Property inspector to create different colored instances of a single symbol.
- 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.
- Use the high quality setting on filters sparingly because it can slow playback.

Bitmap caching

● **Note:** The bitmap is copied to the main Stage as unstretched, unrotated pixels snapped to the nearest pixel boundaries. Pixels are mapped one-to-one with the parent object. If the bounds of the bitmap change, the bitmap is re-created instead of being stretched.

Bitmap caching helps you enhance the performance of nonchanging movie clips in your applications. When you set the `MovieClip.cacheAsBitmap` or `Button.cacheAsBitmap` property to true, Flash Player caches an internal bitmap representation of the movie clip or button instance. This can improve performance for movie clips that contain complex vector content. All of the vector data for a movie clip that has a cached bitmap is drawn to the bitmap, instead of to the main Stage.

Configure a server for Flash Player

For users to view your Flash content on the web, the web server must be properly configured to recognize SWF files.

Your server may already be configured properly. To test server configuration, see TechNote 4151 on the Adobe Flash Support Center at www.adobe.com/go/tn_4151.

Configuring a server establishes the appropriate Multipart Internet Mail Extension (MIME) types so that the server can identify files with the `.swf` extension as Flash files.

A browser that receives the correct MIME type can load the appropriate plug_in, control, or helper application to process and properly display the incoming data. If the MIME type is missing or not properly delivered by the server, the browser might display an error message or a blank window with a puzzle piece icon.

- If your site is established through an Internet service provider (ISP), ask the ISP to add this MIME type to the server:
application/x-shockwave-flash with the .swf extension.
- If you are administering your own server, see your web server documentation for instructions on adding or configuring MIME types.
- Corporate and enterprise system administrators can configure Flash to restrict Flash Player access to resources in the local file system. Create a security configuration file that limits Flash Player functionality on the local system.

Using Adobe Device Central with Flash

Device Central enables Flash users to preview how Flash files will look and function on a variety of mobile devices.

In the past, it was difficult for Adobe Flash Lite developers to test the files they created on mobile devices. Testing content could take a significant amount of time, especially manually exporting and testing on target devices and returning to Flash to make necessary changes.

Device Central is the next generation of mobile emulation and includes new features such as profile updates, memory and performance options, and custom device sets.

Device Central provides mobile content developers and testers with an easy way to create and preview mobile content on a variety of devices. Adobe Device Central displays realistic skins of a wide range of mobile devices that show you what the devices look like and how your content appears on those devices. This enables you to interact with the emulated devices in a way that simulates real-world interactions, including testing different performance levels, memory, battery power levels, and types of lighting.

Device Central provides a library of devices to choose from. Each device has a profile that contains information about the device, including the media and content types it supports (that is, the content that can be used on an individual device such as screen savers, wallpaper, and stand-alone Adobe Flash Player). You can search through available devices, compare multiple devices, and create custom sets of the devices you use most.

Device Central supports different media formats including Adobe Flash, bitmap, video, and web formats. You can use different media formats to create different types of content such as screen savers or wallpaper.

● **Note:** Testing with the Emulator tab cuts the cost and time of testing on mobile devices, but should never replace testing on actual devices. Use Device Central for initial tests as you develop content and then use real devices for final testing.

The Emulator tab in Device Central is designed to simulate content on mobile devices in a realistic way. You can test various media types such as Flash, bitmap, and video, and apply them as different content types, such as stand-alone player, wallpaper, or screen saver.

If you are testing Flash content, for a content file to appear on the Emulator tab on a specific device, the device must support the Flash Lite version and content type that the file uses. For example, if you have a SWF file created in Flash that requires Flash Lite 2 and you try to test the file on the Emulator tab on a device that only supports Flash Lite 1.1, the file does not appear. (In this case, try going to the Available Devices list, group the devices by Flash Lite version, and double-click one of the devices that supports Flash Lite 2.)

To test a Flash Lite document in a device, choose Control > Test Movie.