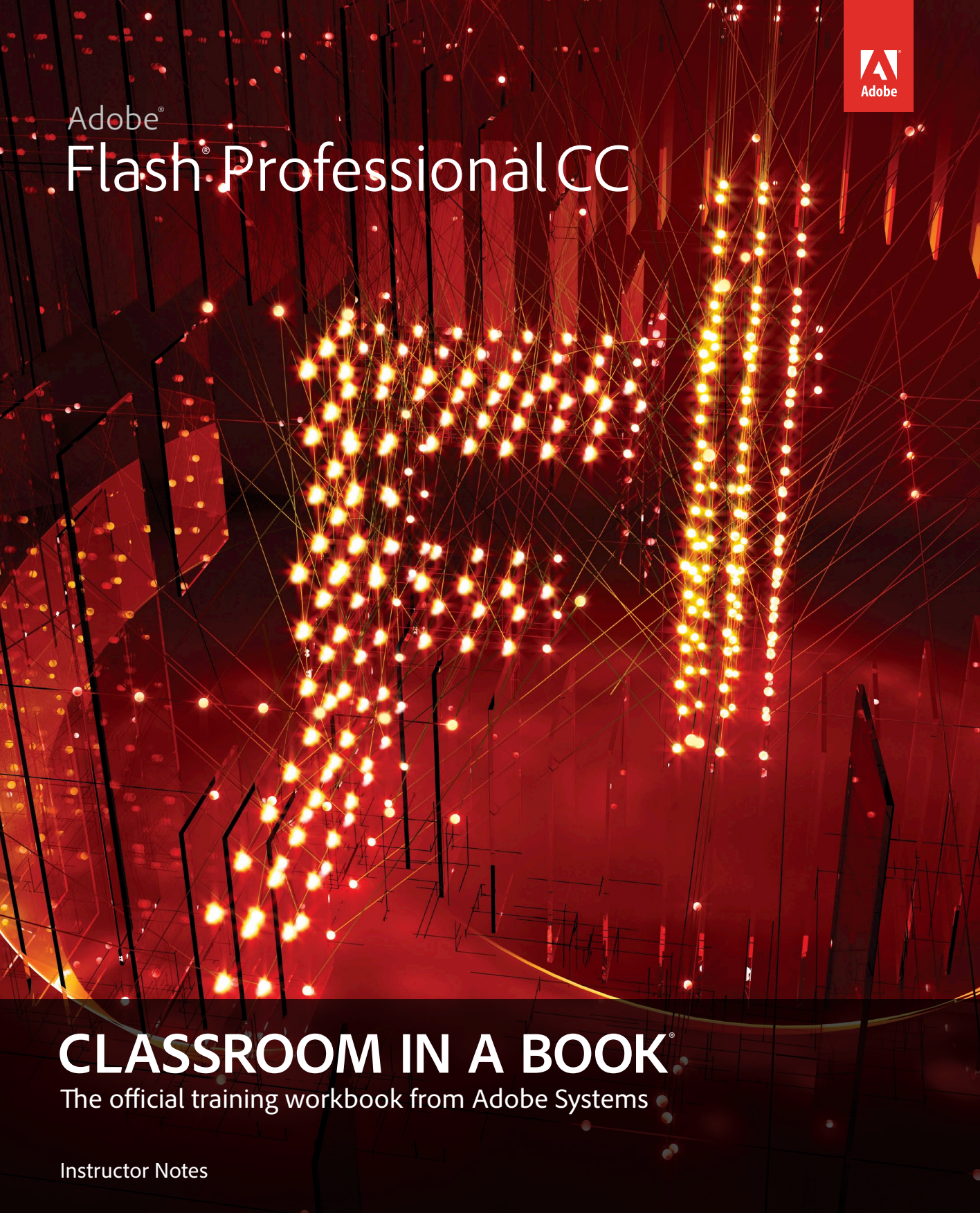
Adobe
Adobe® Flash® Professional CC
CLASSROOM IN A BOOK®
The official training workbook from Adobe Systems
Instructor Notes

Adobe® Flash® Professional CC Classroom in a Book®

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

Adobe Press books are published by Peachpit, a division of Pearson Education located in San Francisco, California. For the latest on Adobe Press books, go to www.adobepress.com. To report errors, please send a note to errata@peachpit.com. For information on getting permission for reprints and excerpts, contact permissions@peachpit.com.

Acquisitions Editor: Rebecca Gulick
Writer: Russell Chun
Development and Copy Editor: Stephen Nathans-Kelly
Production Coordinator: David Van Ness
Compositor: Danielle Foster
Technical Reviewer: Keith Gladstien
Keystroker: H. Paul Robertson
Proofreader: Patricia Pane
Indexer: Valerie Haynes Perry
Cover Designer: Eddie Yuen
Interior Designer: Mimi Heft

Printed and bound in the United States of America

Book
ISBN-13: 978-0-321-92785-9
ISBN-10: 0-321-92785-0

Instructor Notes:
ISBN-13: 978-0-133-43050-9
ISBN-10: 0-133-43050-2

INSTRUCTOR NOTES

Getting Started

The *Adobe Flash Professional Creative Cloud Classroom in a Book* presents students with a series of real-world projects to learn techniques, solutions, and tips using Adobe Flash Professional Creative Cloud software. The instructor notes are intended to complement the information in the *Adobe Flash Professional CC Classroom in a Book*. The lessons in this book require assets, including Flash files that are available for download from the Lesson & Update Files tab on your Account page at www.peachpit.com. Before beginning to work with the lessons, copy the files to the hard drive of each student's computer as described in the Getting Started section of the book.

You may want to use all the lessons in this book, or just 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 Help, Adobe TV, Adobe Design Center, and Adobe Developer Connection, where they can access information such as tutorials, updates, and technical help provided by other Flash users and by Adobe.

Course Strategy

This book contains eleven lessons, so if you're teaching a 15-session course, you can spend extra time on some of the more difficult topics and time-consuming lessons. The lessons that deserve at least two class sessions are Lesson 2 on creating graphics, Lesson 4 on animation, and Lesson 6 on interactive navigation. Depending on the focus of your course, you can spend the remaining extra session on Lesson 5, which covers shape tweening and masking; Lesson 7, which covers video and sound; or Lessons 8 or 9, which covers more advanced ActionScript. If you want to cover new publishing options, such as exporting content to HTML5 and JavaScript, or publishing AIR apps for mobile devices, you can spend more time on Lessons 10 and 11.

For beginners, the lessons that should be covered, at an absolute minimum, are Lesson 1, which acquaints the students to the interface and the concept of the Timeline; Lesson 4, which covers motion tweening; and Lesson 6, which deals with interactivity, the button symbol, and ActionScript.

Enhancements

Flash Professional CC is a completely re-engineered application, which brings dramatic improvements to performance, reliability, and usability. The Flash Professional CC Classroom in a Book integrates many of the enhancements within the lessons. Some of the new features of Flash Professional include:

- Performance improvements, including faster startup times, publishing times, saving times, timeline scrubbing, and reduced CPU usage
- Fresh, modern user interface with simplified Preferences panel
- A more powerful Find and Replace dialog box
- A new Actions panel
- Working in full-screen mode
- Distributing symbols and bitmaps to keyframes
- Swapping multiple symbols or bitmaps
- Improved Photoshop and Illustrator file import
- Smoother, more responsive drawing and editing
- Exporting videos with Adobe Media Encoder
- Resizing the Stage relative to an anchor point
- An integrated Toolkit for CreateJS, which publishes animation to HTML5 and JavaScript
- Comprehensive testing tools, such as device testing via USB and SWF analysis with Adobe Scout
- Synchronizing preferences on multiple machines with Creative Cloud

Streamlined Features

In order to provide a more focused creative environment, Adobe Flash Professional CC has streamlined its feature set. The following is a list of the notable tools that have been dropped from the previous version:

- Support for ActionScript 1 and 2
- TLF Text
- Motion Editor for motion tweens
- Bone tool for inverse kinematics
- Deco Tool
- Project panel
- Printing
- Strings panel
- Behaviors panel
- Object-level undo
- Spelling panel
- Movie Explorer
- Bandwidth profiler in Test Movie mode
- FXG file import or export
- Kuler panel
- Video Cue Points (but still available in Media Encoder and through ActionScript)
- Closed Captioning
- Video playback on Stage with FLV playback component
- Device Central
- Importing SWFs
- PicWhip in the Code Snippets panel
- Auto-Save
- File Info (XMP Metadata)
- Importing some bitmap formats (BMP, TIFF, AutoCad) and some sound formats (AIFF, Sound Designer, Around AU, Adobe Sounds Document)
- Publishing projectors

Lesson 1: Getting Acquainted

Lesson Overview

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

Starting Flash and opening a file

Demonstrate two ways of opening a document in Flash: by double-clicking the Flash document or by starting Flash, choosing File > Open. Let your students know that a Flash file can either have the extension FLA or XFL. The XFL document is a newer Flash format (requiring CS5 or later), which is contained within a folder of other related assets. Explain the difference between a published SWF file and the FLA or XFL file. A SWF file is a compressed, published Flash document, which is uploaded to a server to play with the Flash Player in a browser, whereas the FLA file 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, Properties inspector, and the Edit bar above the Stage. Explain to 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 other Adobe applications installed, you can show students the similarities between panels in Flash Professional CC and panels in other Adobe Creative Cloud applications. The Timeline is similar to the Timeline in Adobe Premiere Pro and Adobe After Effects. The Properties inspector is similar to the one in Dreamweaver. Many commands work across applications, which will ease the anxiety of learning a new program. For example, copy and paste, or holding down the Shift key to select multiple items, behave the same in Flash.

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 need 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 content 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 Properties inspector

Demonstrate that the options in the Properties inspector change to reflect what's currently selected. The Properties 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 how 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 students that additional options for a selected tool are also available in the Properties inspector.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. What is the difference between the *.fla file, the *xfl file, and the *.swf file, and how do you create them?
2. What do you do when you can't find a panel that you know exists?
3. How does Flash measure its X and Y coordinates on the Stage?
4. How do you change the dimensions of the Stage and keep the contents proportional?
5. If your panels have been moved out of their normal positions, how do you put them back to their default positions?
6. How do you change the appearance of the Timeline?

Answers

1. The *.fla and the *.xfl are both editable source files for a Flash project. When you save a Flash project, you can save it as a *.fla document or as an uncompressed *.xfl document, which exposes the contents of the Flash file for other developers. The *.swf is the exported file that is generated when you publish a movie for the Flash Player or test a movie (Control > Test Movie > in Flash Professional). The *.swf is the file that is uploaded to your Web server to play in the Flash Player in a browser.
2. You can access all the panels by choosing Window from the top menu.
3. The X and Y values are measured on the Stage from the top-left corner. X begins at 0 and increases to the right, and Y begins at 0 and increases downward.
4. In the Properties inspector, click the Edit button next to Size. In the Document Settings dialog box that appears, enter new pixel values in the Stage size fields and choose Scale content. The Anchor option lets you choose the origin from which your content is resized, if the proportions of the new Stage are different.
5. Choose Window > Workspaces > Reset Essentials to reset the Essentials workspace, which is the default arrangement of panels.
6. When you want to see more layers, select Short from the Frame View pop-up menu in the upper-right corner of the Timeline. The Short option decreases the height of frame cell rows. The Preview and Preview in Context options display thumbnail versions of the contents of your keyframes in the Timeline. You can also change the width of the frame cells by selecting Tiny, Small, Normal, Medium, or Large.

Lesson 2: Working with Graphics

Lesson Overview

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 showing a mug of steaming coffee over a wavy background with pieces of text that are aligned to each other. 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, align objects, create gradients and transparencies, create curves, and manipulate objects with the Free Transform tool. They also learn the difference between drawing modes in Flash. Be sure to introduce the Convert to Bitmap feature, which can be helpful to simplify artwork and reduce the performance load on mobile devices.

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 students how to select the fill or the stroke of a shape by clicking on it, or to select the entire shape (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 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 mention 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 in the Tools panel, under Fill Color options, in the Alpha field.

Creating gradients

In a gradient, one color gradually changes into another. Flash can create linear gradients, which change color horizontally, vertically, or diagonally; or radial gradients, which change color moving outward from a central focal point. Encourage students to apply a radial gradient and a linear gradient in succession to the same shape to see how the gradients differ. Students can add color pointers beneath the gradient definition bar in the Color panel to add up to 15 color transitions. In addition to choosing colors and positioning the color pointers for a gradient, demonstrate how you can adjust the size, direction, or center of a gradient fill with the Gradient Transform tool.

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 Classic text, which is used for text used for display purposes only. Input and Dynamic text, the two other forms of text, are used when you want to control the text with ActionScript. To format text, select it with the Text tool, and then select formatting options in the Properties inspector. To manipulate a text box as an object, select it with the Selection tool and make changes in the Properties inspector or Transform panel, or simply drag the text box handles to resize it. Resizing the text box does not resize the text, but it does reflow the text in the text box.

Note that TLF text (Text Layout Framework) has been dropped from Flash Professional Creative Cloud. If students open an .fla file containing TLF text in Flash Professional CC, Flash converts the text to Static text.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. How do you create a corner point when editing the contours of a shape with the Selection tool?
2. What are two ways you can change the fill of a shape?
3. What's the difference between a linear gradient and a radial gradient?
4. What is the function of the Lock Fill option for the Paint Bucket tool?

Answers

1. Hold down the Alt (Windows) or Option (Mac) key while dragging the sides of a shape to add a new corner.
2. One way you can change the fill of a shape is to choose the Paint Bucket tool, then choose a new Fill color in the Properties inspector, and then click on the shape. The other way is to select the fill on the Stage, and then choose a new color in the Properties inspector.
3. A linear gradient changes color horizontally, vertically, or diagonally, whereas a radial gradient changes color moving outward from a central focal point.
4. The Lock Fill option locks the current gradient to the first shape where it was applied so that subsequent shapes extend the gradient.

Lesson 3: Creating and Editing Symbols

Lesson Overview

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. Flash Pro CC provides an updated AI Importer that includes a great degree of control in determining how Illustrator artwork is imported into Flash.

Flash Professional automatically recognizes layers, frames, and symbols. If students are more familiar with Illustrator, you can suggest that students design layouts in Illustrator, and then import them into Flash to add animation and interactivity.

Choose File > Import > Import To Stage or File > Import > Import To Library to import the artwork into Flash. Alternatively, students can even copy artwork from Illustrator and paste it into a Flash document.

Importing Layers

When an imported Illustrator file contains layers, you can import them in any of the following ways:

- Convert Illustrator layers to Flash layers
- Convert Illustrator layers to Flash keyframes
- Convert all Illustrator graphics to a bitmap
- Convert all Illustrator layers to a single Flash layer

Importing Symbols

Working with symbols in Illustrator is similar to working with them in Flash. In fact, you can use many of the same symbol keyboard shortcuts in both Illustrator and Flash: Press F8 in either application to create a symbol. When you create a symbol in Illustrator, the Symbol Options dialog box lets you name the symbol and set options specific to Flash, including the symbol type (such as movie clip) and registration point location.

If you want to edit a symbol in Illustrator without disturbing anything else, double-click the symbol to edit it in isolation mode. Illustrator dims all other objects on the artboard. When you exit isolation mode, the symbol in the Symbols panel—and all instances of the symbol—are updated accordingly.

Use the Symbols panel or the Control panel in Illustrator to assign names to symbol instances, break links between symbols and instances, swap a symbol instance with another symbol, or create a copy of the symbol.

Copying and Pasting Artwork

When you copy and paste (or drag and drop) artwork between Illustrator and Flash, the Paste dialog box appears. The Paste dialog box provides import settings for the Illustrator file you're copying. You can paste the file as a single bitmap object, or you can paste it using the current preferences for AI files. Just as when you import the file to the Stage or the Library panel, when you paste Illustrator artwork, you can convert Illustrator layers to Flash layers.

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. It's also useful to talk about symbols as containers. Like containers, symbols hold different kinds of content—whether it's an imported Illustrator graphic, a bitmap, or a piece of text. You can always edit or replace the contents of the symbol, just like you can edit or replace the contents of a container.

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 Properties 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, because there is really no "conversion." Explain that the selected items are simply placed inside 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 students the two ways in which you can edit your symbols. First, you can double-click the symbol in the Library panel and edit its contents in symbol-editing mode. Return to the main Timeline by clicking the Scene 1 button on the Edit bar above the Stage. Second, 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 Properties 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 students to explore the different Blend options. Filters are special effects, such as blurs and glows, which you can apply to instances. Additionally, you can position or rotate any instance in three-dimensional space using the 3D Translation and 3D Rotation tool.

The Visible and Render options in the Properties inspector can help you manage movie-clip instances. The Visible option is an easy way to position an instance on the Stage and control its visibility at a later point in time, either with ActionScript or with the Visible option. The Render options can help performance on mobile devices by simplifying the display of complicated vector art as bitmap art.

Using rulers, guides, and grids

Explain to students that many tools can help them position instances and other objects on the Stage. In Lesson 1, students used the Properties inspector to position objects by changing the X and Y coordinates, and in Lesson 2, they learned to use the Align panel to align several objects to each other. Another way to position objects on the Stage is to use rulers, guides, and grids. Rulers (View > Rulers) appear on the top and left edge of the Pasteboard to provide measurement along the horizontal and vertical axes. Guides (View > Guides > Show Guides) are vertical or horizontal lines that you can pull out from the rulers and position on the Stage but do not appear in the final published movie. The grid (View > Grid > Show Grid) superimposes vertical and horizontal lines on the Stage and also does not appear in the final published movie. Another useful feature is the Move Guide option, which appears when you double-click on a guide. In the Move Guide dialog box, you can enter pixel values to position a guide exactly where you want it.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. What are the three kinds of symbols, and how do they differ?
2. When you import a Photoshop document into Flash, how do you preserve the layer effects?
3. How do you convert a bitmap image into a vector graphic?
4. What are blending effects?
5. What is the difference between the Global and Local Transform option for the 3D Translation and 3D Rotation tool?
6. How do you reset the transformations of an instance on the Stage?

Answers

1. The three symbols are Graphic, Button, and Movie Clip. A Graphic symbol is the most basic of symbols, which you can use for animation, but does not support ActionScript, blending effects, or filters. A Button symbol is used for interactivity and contains four special keyframes that describe how it appears when the mouse is interacting with it. A Movie Clip symbol contains its own independent Timeline. You can apply filters, color settings, and blending modes to a Movie Clip instance, and you can use ActionScript to control a Movie Clip instance.
2. In the Import to Stage dialog box, For Layer conversion, choose the Maintain editable paths and effects option. Any layer effects such as transparencies or blending will be preserved.
3. To convert a bitmap to a vector, import the bitmap image into Flash. Select the bitmap and choose Modify > Bitmap > Trace Bitmap. The options in the dialog box that appears determine how faithful of a trace the vector image will be to the original bitmap.
4. Blending refers to how the colors of an instance interact with the colors below it.
5. When the Global Transform option is selected, rotation and positioning is relative to the global, or Stage, coordinate system. The 3D display over the object that you're moving shows the three axes in constant position, no matter how the object is rotated or moved. When the Global Transform option is not selected, rotation and positioning is relative to the object. The 3D display shows the three axes oriented relative to the object, not the Stage.
6. If you want to reset the transformations of your instance, you can use the Transform panel (Window > Transform). Click the Remove Transform button in the lower-right corner of the Transform panel to reset the instance to its original settings.

Lesson 4: Animating Symbols

Lesson Overview

Lesson 4 uses an animated motion picture promotional Web site to introduce students to motion tweens, easing, editing the paths of motion, frame-by-frame animations, 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 involving other properties other than an object's location. 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 its path of motion, and its easing, which is the way an object accelerates or decelerates.

The basic workflow for creating motion 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 change one of its properties such as its position, transparency, or size.

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. It's helpful to think of Motion tweening as "instance tweening," since you are changing the properties of an instance over time. This will help students differentiate between Motion tweening and Shape tweening, which they will encounter in the next lesson.

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 Ctrl-click (Windows) or Command-click (Mac) to select the keyframe, and then click and drag it to a new position.

Demonstrate to 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.

Span Based vs. Frame Based Selections

If students are having trouble selecting and editing their keyframes, double-check their preferences on span-based or frame-based selections. By default, Flash does not use span-based selection, which means you can select individual keyframes within a motion tween. However, if you prefer to click on a motion tween and have the entire span (the beginning and end keyframes, and all the frames in between) be selected, you can enable span-based selection from the options menu on the top-right corner of the Timeline.

With span-based selection enabled, you can click anywhere within the motion tween to select it, and move the whole animation backward or forward along the Timeline as a single unit.

If you want to select individual keyframes while span-based selection is enabled, hold down the Ctrl (Windows)/Command (Mac) key and click on a keyframe.

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, students create a nested animation of the alien waving his arms and another nested animation of the car jerking up and down. Ask your students to brainstorm what other kind of nested animations would require an animation inside a movie-clip symbol.

For nested animations, demonstrate the export functions that generate a PNG sequence or a sprite sheet. PNG sequences and sprite sheets are used as animation assets for alternative (non-Flash) ways to create animation.

Frame-by-frame animation

Frame-by-frame animation refers to the illusion of movement created by seeing the incremental changes between every keyframe. It's the closest to traditional hand-drawn cel animation, and it's as tedious. In Flash, you can change a drawing in every keyframe, and create a frame-by-frame animation. In this lesson, students insert a frame-by-frame animation inside a movie-clip symbol of a car to make it move up and down in a jittery fashion. When the movie clip loops, the car will rumble slightly to simulate the idle of the motor.

Ask students what situations would require frame-by-frame animations. Remind students that frame-by-frame animations increase your file size rapidly because Flash has to store the contents for each keyframe, so warn them to use the technique sparingly.

Motion Editor

Users of the older versions of Flash Pro will immediately notice that the Motion Editor panel is gone in this release. Flash Professional Creative Cloud dropped the Motion Editor, which displayed motion tweens as graphs for each object's property.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. Why is animation referred to as "tweening"?
2. If you want to animate an airplane moving across a static background, how many layers are required?
3. Identify two ways that you can lengthen the duration of a motion tween.
4. What is the Motion Presets panel, and how do you use it?
5. What does the Orient to path option do?
6. What's the difference between an ease-in and an ease-out?

Answers

1. The term "tweening" comes from the world of classic animation. Senior animators would be responsible for drawing the beginning and ending poses for their characters. The beginning and ending poses were the keyframes of the animation. Junior animators would then come in and draw the "in-between" frames, or do the "in-betweening." Hence, "tweening" refers to the smooth transitions between keyframes.
2. You need two layers: one layer to create a motion tween for the airplane and a second layer for the static background.
3. To lengthen the duration of a motion tween, you can drag the end of a tween span to lengthen its total time. You can also choose Insert > Timeline > Frame (F5) to add frames to a tween span.
4. The Motion Presets panel (Window > Motion Presets) stores particular motion tweens so you can apply them to different instances on the Stage. To save a motion tween, select the motion tween on the Timeline or the instance on the Stage. In the Motion Presets panel, click the Save selection as preset button and name your motion tween to be stored in the Motion Presets panel.
5. The Orient to path option in the Properties inspector adjusts the object traveling along its path so it is oriented in the direction in which it is heading.
6. Easing refers to the way in which a motion tween proceeds. An ease-in refers to motion that gradually begins from its starting keyframe. An ease-out refers to motion that gradually slows down as it approaches its ending keyframe.

Lesson 5: Animating Shapes and Using Masks

Lesson Overview

Lesson 5 introduces shape tweening and masking. Shape tweening allows students to morph one shape to another, and is ideal for creating organic animations. Masking is a way to selectively show only parts of a layer. Students explore both techniques in combination to create more sophisticated effects.

Animating shapes

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. Explore with your students different animation challenges and have them identify which techniques would be most appropriate to complete the animation. In this lesson, students create a simple animated logo that contains an animation of a flickering flame.

Emphasize to 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.

Students should be familiar with the basic workflow: Create a shape in the initial keyframe, insert a new keyframe further down the Timeline, change the shape in the newly created keyframe, then apply a shape tween between the two keyframes. Create a broken shape tween by clearing the last keyframe, and help them recognize the broken shape tween. Identify ways to fix the tween.

Changing the pace

If students are comfortable editing the pacing and timing of motion tweens, then they should have no trouble doing the same with shape tweens. Reinforce that the process is identical. If you want the animation to proceed quicker, then the time between keyframes must be shorter. You can select individual keyframes and move them closer or farther apart to change the pacing of the shape tween.

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; for best results, place shape hints in a clockwise or counterclockwise fashion.

Previewing the Animation with Onion Skin Outlines

It's sometimes useful to see how your shapes are changing from one keyframe to another on the Stage all at once. Seeing how the shapes gradually change lets you make smarter adjustments to your animation. You can do so using the Onion Skin Outlines option, available at the bottom of the Timeline.

Onion Skin Outlines show the contents of the frames before and after the currently selected frame.

The term "onion skin" comes from the world of traditional hand-drawn animation, when animators would draw on thin, semi-transparent, tracing paper known as onionskin. When creating an action sequence, animators flip back and forth quickly between drawings held between their fingers. This allows them to see how the drawings smoothly connect to each other.

To use Onion Skin Outlines, click the Onion Skin Outlines button to enable it. Drag the beginning or end marker to select the range of frames around the playhead that you want to show. You can also select the preset marker options in the Modify Markers menu.

Only the content in the keyframes of a shape tween are displayed fully rendered. All other frames are shown as outlines. To see all frames fully rendered, click the Onion Skin option.

Creating masks

Masking is a way of selectively hiding and displaying content on a layer. Masking allows 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, and you can animate both the shape of the spotlight or the image under the spotlight.

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. Both the mask and masked layers must be locked in order for you to see the results of the mask in the authoring environment.

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.

For the animated logo students create in this lesson, they add a mask and an animation in the masked layer that makes the text a little more visually interesting.

Traditional Masks

It might seem counterintuitive that the shapes in the Mask layer reveal, rather than hide, the content in the Masked layer. However, that's exactly how a traditional mask in photography or painting works. When a painter uses a mask, the mask protects the painting from paint splatters. When a photographer uses a mask in the darkroom, the mask protects the photo-sensitive paper from the light, to prevent those areas from getting any darker. So thinking of a mask as something that protects the lower, Masked, layer, is a good way to remember which areas are hidden and which areas are revealed.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. Of the following properties, which one CANNOT be smoothly animated in a shape tween: gradient fill, stroke size, width, transparent fill, blur filter?
2. How do you remove shape hints?
3. What is the difference between using an opaque shape and a shape with a fill at 50% opacity in the Mask layer?
4. A mask requires two layers: a Mask layer, and a Masked layer. Which one appears above the other on the Timeline?

Answers

1. The blur filter cannot be smoothly animated with a shape tween. Shape tweens require shapes, and filters are only applied to symbol instances.
2. To remove a shape hint, drag it entirely off the Stage and Pasteboard. To remove all shape hints, choose Modify > Shape > Remove All Hints. Removing a shape hint in one keyframe will remove its corresponding shape hint in the other keyframe.
3. There is no difference; Flash does not recognize different Alpha levels in a mask, so a semi-transparent fill in the Mask layer has the same effect as an opaque fill, and edges will always be hard-edged. However, with ActionScript you can dynamically create masks that will allow transparencies.
4. The Mask layer appears above the Masked layer on the Timeline. The Masked layer is also indented, indicating that it is affected by the Mask layer.

Lesson 6: Creating Interactive Navigation

Lesson Overview

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. Students learn about basic event handling and the commands for controlling the playhead. The end result is an interactive restaurant guide in which the user can click on any thumbnail image of a restaurant to see more information and reviews of that restaurant.

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 jump around and go to different frames of the Timeline based on which button the user clicks. Different frames on the Timeline contain different content. Users don'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.

Emphasize that there are many other ways to interact with Flash content: clicking a button is the simplest and is the one demonstrated in this lesson. You can also interact with content with mobile gestures such as pinching or swiping, or by tilting a mobile device.

Creating buttons

The buttons in this lesson include small thumbnail bitmap images with some text. 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 interacting 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 panel. The names in the Library panel are simple organizational reminders.

Instance naming follows these simple rules:

- Do not use spaces or special punctuation. Underscores are OK to use.
- Do not begin the name with a number.
- Do not use any words that are also ActionScript key words, or commands.
- Be mindful of uppercase and lowercase letters.

One source of confusion is the optional, but helpful, _btn suffix. Some students think that the suffix is only required in the Instance Name field in the Properties inspector but not in the actual ActionScript coding, or vice versa. Remind them that the names must match exactly.

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 Professional CC uses ActionScript 3.0, the scripting language that extends Flash functionality and enables interactivity in Flash projects. Previous versions, ActionScript 1.0 and 2.0, are not supported. Students who haven't worked with scripting languages before may initially be intimidated and confused by ActionScript, which is an object-oriented scripting language. 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, includes 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, tell students that they can always refer to *Programming ActionScript 3.0* or the *ActionScript Language References*; both are available in Flash Help. It's particularly important for students to be able to find and learn how to use an ActionScript element on their own by reading the Help entries.

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 ways of incorporating ActionScript into your Flash document files, there are several different ways to learn ActionScript.

Dealing with errors

All students struggle with syntax errors when they learn ActionScript. Explain to your students that this is a normal process, and even veteran developers deal with debugging. Share with them the ways in which they can identify and fix the mistakes in their code:

- Use the errors that appear in the Compiler Errors panel to identify the kind and source of the problem.
- Use the color-coding of the code as it appears in the Script pane to check if the correct keywords are colored.
- More advanced students can begin to use the trace() command to send messages to the Output panel.

Creating event handlers for buttons

Events happen in the Flash environment that Flash can detect and respond to. Explain the concept of an event to 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. Other mobile device interactions are also events, such as pinching, swiping, or even accessing the accelerometer by tilting the device. 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 responds to them with 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 students which of the words are placeholders for objects and parameters for their situation. The phrase 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):void{ };
```

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.

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

- MouseEvent.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 each from ActionScript easier.

A few essential points for students to remember: Use straight double quotes around the name of the frame label in their ActionScript code. If they want to use the gotoAndPlay() command, 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.

Using the Code Snippets panel

The Code Snippets panel provides commonly used ActionScript code for you to easily add simple interactivity to your Flash project. If students are unsure of coding their own buttons, they can use the Code Snippets panel to learn how to add interactivity. You can also save, import, and share code between a team of developers to make the development and production process more efficient. In this lesson, students use the Code Snippets panel to add interactivity to their home button.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. What happens if a button symbol has an empty keyframe in its Up state but filled keyframes in its Over, Down, and Hit states?
2. What is the significance of the color of a shape in a button symbol's Hit keyframe?
3. In what kind of situation would you want a button's Hit state to be larger than its Up state?
4. What is wrong with the instance name of this button, 02restaurantbtn, and how would you fix it?
5. What additional information is needed in between the parentheses of the command gotoAndStop()? What information is needed in between the parentheses of the command stop()?
6. How do you use the Code Snippets panel to program a button to go to a specific frame?

Answers

1. If a button symbol has an empty keyframe in its Up state, the button instance on the Stage is invisible. The mouse still can interact with the button instance, but the user won't be able to see it.
2. There is no significance to the color of a shape in a button symbol's Hit keyframe. Only the size and position of a shape in the Hit state determines the clickable area.
3. You may want the clickable area to cover a larger area than the visible button. For example, a typical pause button on consumer electronics is represented by two parallel vertical lines. You would want the Hit state to cover the gap between the vertical lines so users have an easier time clicking the button.
4. The instance name cannot begin with a number. You may want to add an underscore before btn, but it is not incorrect without it.

5 The gotoAndStop() command requires the destination frame number or name of the frame label, in quotation marks. The stop() command doesn't require any information between its parentheses.

6 Choose Window > Code Snippets, or if your Actions panel is already open, click the Code Snippets button at the top right of the Actions panel to open the Code Snippets panel. The code snippets are organized in folders that describe their function. First, select a button instance on the Stage. In the Code Snippets panel, expand the folder called Timeline Navigation and double-click the Click to Go To Frame and Stop option. You can also click the Add to current frame button. Flash automatically adds the ActionScript code to the Timeline. Customize the ActionScript code by providing the correct destination frame in the gotoAndStop() command.

Lesson 7: Working with Sound and Video

Lesson Overview

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 Adobe Media Encoder to encode their video files to the correct format, and integrate video into their Flash project. Students also learn to use Media Encoder to export Flash content to video in a variety of formats.

Using sounds

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

Selecting the appropriate option from the Sync 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.

- **Stream** synchronizes the sound for playing on a Web site. 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 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.

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.

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 Properties 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 start working with video, consider what video quality you need, what video format to use, what size can accommodate your layout, 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.

Using Adobe Media Encoder

You can convert your video files into the FLV or F4V format using Adobe Media Encoder, a standalone application that comes with Flash Pro CC. Adobe Media Encoder can convert single files or multiple files (known as batch processing) to make your workflow 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 the Preset selection in the display window, or choose Edit > Export Settings. The Export Settings dialog box appears.

To simplify the conversion process, you can rely on the preset settings, which offer common encoding profiles for a one-step operation.

Playback of external video

To play back an external FLV or F4V file, in Flash, 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 video files provides the following capabilities that are not available when using embedded video:

- **Long video clips.** You can use longer video clips without slowing down playback. External video 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.
- **Different frame rate.** An external video 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.
- **No interruption.** With external video 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). Video files can perform functions independently of the Flash document, so they do not interrupt playback.

- **Captions and cue points.** Captioning video content is easier with external video files because you can use callback functions to access metadata for the video. Cue points are special markers that can be tied to ActionScript code for additional functionality.

Embedding Flash Video

Embedded video lets you insert a video file directly 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:

- **Performance problems.** 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.
- **Synchronization issues.** 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.
- **Maximum length.** There is a maximum length of 16,000 frames for embedded movies.

Despite these limitations, embedded video has a place for small, well-placed video segments.

Exporting video

You can use Adobe Media Encoder to export your Flash content to video.

For example, use the powerful drawing and animation tools within Flash to create animation, and then export it as an HD broadcast-quality video, or for playback on a variety of platforms such as an iPhone, Nook, Kindle, or Android device. Sound on the Timeline and even nested animations are exported.

In this lesson, students export the animation they created in Lesson 4, the cinematic splash screen for the fictional motion picture "Double Identity."

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. How do you set the quality of sounds?
2. How do you fade out the volume of a sound?
3. What are the Sync options for sound?
4. Which video formats are compatible with Flash?
5. Which of these options are *not* available in Adobe Media Encoder: (1) Cropping a video; (2) Trimming the length of a video; (3) Combining multiple video clips together; (4) Stripping the audio from a video; (5) Changing the frame rate of a video; (6) Adding cue points to video?
6. What is the SkinAutoHide option, and how do you enable it?
7. If your video has a transparent background, what is the required format for your video?

Answers

1. Set the sound quality and compression in the Publish Settings options. Select the Flash format from the left and click on the Audio Stream and Audio Event values. In the Sound Settings dialog box that appears, change the Bit rate, which determines the quality of the sound in your final, exported Flash movie. The higher the bit rate, the better the quality.
2. You can change the volume levels in the Edit Envelope dialog box. Select the first keyframe that contains your sound and, in the Properties inspector, click the Pencil button. The Edit Envelope dialog box appears. Click on the top horizontal line of the waveform to add keyframes. Drag the keyframes down to the bottom of the window to reduce the volume at that point in time. You can also select the Fade-out option from the Effect menu in the Properties inspector to apply a global fade from start to finish.
3. Sound sync refers to the way the sound is triggered and played. There are several options: Event, Start, Stop, and Stream. Stream ties the sound to the Timeline so you can easily synchronize animated elements to the sound. Event and Start are used to trigger a sound when the playhead enters the keyframe. Event and Start are similar except that the Start sync does not trigger the sound if it is already playing (so there are no overlapping sounds possible with Start sync). The Stop option is used to stop a sound.
4. The video formats that are compatible with Flash are FLV, F4V, and video encoded with the H.264 codec.

5 Option three. You cannot use Adobe Media Encoder to combine multiple video clips together.

6 To hide the video controller and only display it when users roll their mouse cursor over the video, use the SkinAutoHide option. Select your video on the Stage, and enable the SkinAutoHide option in the Component Parameters section of the Properties inspector.

7 If you want to preserve a transparent background, your video must be encoded as an FLV with the On2 VP6 codec.

Lesson 8: Loading and Displaying External Content

Lesson Overview

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. 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. You can either write the code in the Actions panel yourself or use the Code Snippets panel to help. A ProLoader object must first be created, as in the following statement:

```
var myProLoader:ProLoader=new ProLoader();
```

Next, the load() method should be called for the ProLoader object. The parameter for the load() method is the URLRequest object, which defines the external SWF file. The URLRequest object can also define a JPEG image, if you want to load an image instead of a Flash file.

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

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

Then, to display the loaded SWF on the main Stage, you need to add the ProLoader object to the display list with the following command:

```
addChild(myProLoader);
```

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

Explain to students that there are two different ways to remove the loaded content. One way is to remove the ProLoader object from the Stage with removeChild(). The second way is to unload the loaded content with unload(). Make sure they know the difference between these two approaches.

● **Note:** For those familiar with ActionScript, the ProLoader object is the newer version of the Loader object. The methods and application of the ProLoader are identical to the Loader object, but the ProLoader object handles the loading of SWF files with external libraries more consistently and reliably.

Controlling movie clips

The initial animations are nested inside individual movie clips, and students learn to control each movie-clip's Timeline. 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.

Positioning of loaded content

Loaded content is aligned with the Stage (or whatever it is loaded into). The registration point of the ProLoader object is at its top-left corner, so the top-left corner of the external SWFs aligns with the top-left corner of the Stage (where x=0 and y=0). Since the four external Flash files (page1.swf, page2.swf, page3.swf, and page4.swf) all have the same Stage size as the Flash file that loads them, the Stage is completely covered.

For advanced students, you can introduce ways to position the loaded content. If you want to place the ProLoader object in a different horizontal position, you can set a new X value for the ProLoader object with ActionScript. If you want to place the ProLoader in a different vertical position, you can set a new Y value for the ProLoader. In the Actions panel, enter the name of the ProLoader object, followed by a period, the property x or y, and then the equal symbol and a new value. In the next lesson, students will work more with movie-clip properties such as x, y, scaleX, scaleY, width, and height.

Managing overlapping content

In this lesson, there is only one ProLoader object that loads and displays a single external SWF at a time that covers the whole Stage. However, when you use multiple ProLoader objects to load several different external files or images at time, you often have to manage overlapping content.

Juggling how different pieces of content overlap each other depends on their depth level in the Display List, which is a list that organizes all the visible content. The Display List manages the order of visible items with a simple number, called an index, starting at 0. Items that are higher on the list overlap objects that are lower on the list.

If students are interested, consider explaining a few ways to modify the order of objects in the Display List. You can use the command, addChildAt(), which takes two arguments inside of its parentheses. The first argument is the object you want to place, and the second is the index in the Display List. You can also just use the command, setChildIndex(), if the object is already in the Display List.

Another approach is to swap the depth levels of two objects using swapChildren(). Provide two objects as the arguments and Flash switches their order in the Display List.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. What is the difference between using removeChild() and unload() to remove the loaded content?
2. How do you position the Loader object on the Stage?

Answers

1. If you use removeChild(), the ProLoader object (along with its content) on the Stage is removed from the display. If you use unload(), the ProLoader object remains on the Stage, but any content that has been loaded into it is removed.
2. You can position the ProLoader object by assigning a new value to its *x* or *y* properties with ActionScript. In the Actions panel, enter the name of the ProLoader object, followed by a period, the property *x* or *y*, an equal symbol, and a new value.

Lesson 9: Using Variables and Controlling Visual Properties

Lesson Overview

In Lesson 9, students add ActionScript to create interactivity that dynamically change the graphics on the Stage based on the movement of their mouse cursor. All the content—the images, text, and accompanying graphics—are already included in the Library panel and ready for the students to use.

This lesson introduces students to more complex mouse interactions –when the mouse cursor moves over an object, when the mouse cursor moves off an object, and when the mouse cursor simply moves. Students tie these events to changes in the appearance of graphics on the Stage, so the visual feedback is immediate.

Visual properties of movie clips

In this lesson, students learn to change the appearance of a movie clip purely with ActionScript. You can change an object's rotation, width, height, transparency, and many other visual properties just with code. This allows you to change the appearance of movie-clip instances to respond dynamically so you can make more sophisticated and more interactive environments.

To change the appearance of a movie-clip instance with ActionScript, you first enter the name of the object you want to target, then type a period (a dot), the property name, and then an equal sign. To the right of the equal sign, enter a value. The equal sign assigns the value to the property. For example:

thumbnail.rotation=45;

In this statement, the object named **thumbnail** is targeted, and its **rotation** property is assigned a value of 45. As a result, Flash rotates the thumbnail object 45 degrees clockwise.

Remind students that as they learn the properties for each kind of object, they must also learn the data type for those properties. Some properties take Number data for degrees, while others may use Number data for pixels. Some properties take String values (characters) for names, and others use Boolean values (true or false).

The following is an essential list of movie-clip properties and their data types that you should have your students memorize:

BASIC PROPERTIES OF THE MOVIE CLIP	
x	Horizontal position, in pixels (Number data)
y	Vertical position, in pixels (Number data)
rotation	Angle, in degrees clockwise from the vertical (Number data)
alpha	Transparency, from 0 (transparent) to 1 (opaque) (Number data)
width	Width, in pixels (Number data)
height	Height, in pixels (Number data)
scaleX	Percentage of horizontal scale of original movie clip, 1=100% (Number data)
scaleY	Percentage of vertical scale of original movie clip, 1=100% (Number data)
visible	Visibility, false=invisible, true=visible (Boolean data)
name	Instance name (String data)

Mapping mouse movements to visual changes

The essential idea for the interactivity in this lesson is to map one set of values—the position of the mouse cursor—to another set of values—the position of the large image. This is a useful, and widespread, approach that you see in many different interfaces. Brainstorm with your students where this kind of mapping can be found. For example, a basic volume controller maps the position of a knob on a slider to the volume level of sound. A scroll bar maps the position of the scroll button to the position of text on a Web page. Google Maps correlates the position of the zoom slider to the scale factor of the map.

For this project, students use the property mouseX and mouseY to get the x and y position of the mouse cursor and variables to store information.

Variables and data types

Variables are simply containers for data. When you declare a variable, ActionScript 3.0 requires that your variables be "strictly typed." That means ActionScript needs you to tell it what kind of data the variable stores. You can't mix data types. For example, if you create a variable for Number data, you can't assign characters (String data) to the variable.

Explain the basic data types: Number, String, and Boolean.

- Number data includes any kind of number such as a negative number, positive number, or decimals. There are also more specific data types for numbers, such as int (for integers) and uint (for unsigned integers).
- String data includes any sequence of characters, such as a password, or a Web address. String data are enclosed in quotation marks.
- Boolean data is either true or false, and you do not use quotation marks around those terms.

Output panel

As students learn more about ActionScript code, it's important that you explain ways to debug code and provide ways to track variables. The `trace()` statement is useful for sending information to the Output panel when your Flash project is in Test Movie mode. The `trace()` statement displays expressions within its parentheses, so you can insert the statement in your code when you need an alert and information on the current values of your variables.

Object registration and the coordinate space

With all the calculations to position objects on the Stage, make sure your students understand the coordinate space and the registration points of their objects.

When you create your movie-clip symbols, you choose the position of the registration point. Most commonly, the registration point is set at the top-left corner or the center of your object.

All measurements, such as the x-value and y-value, are made from that registration point. All transformations, such as rotations or scaling, are made from the registration point as well.

The registration point for the Stage is at the top-left corner. The x-values increase to the right, and the y-values increase to the bottom.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. If you want to make a custom graphic follow the mouse cursor, you would add an event listener to what mouse event?
2. The x and y properties of a movie clip require what kind of data type?
3. What ActionScript statement sends information to the Output panel, and why is that useful?

Answers

1. You would add an event listener for MOUSE_MOVE in order to track the position of the mouse cursor, and make a custom graphic follow the cursor.
2. The x and y properties of a movie clip require Number data, and they represent the horizontal and vertical distance, in pixels, on the Stage.
3. The `trace()` statement displays the expressions within its parentheses in the Output panel during Test Movie mode. You can insert the statement in your code when you need an alert and information on the current values of your variables.

Lesson 10: Publishing to HTML5

Lesson Overview

In Lesson 10, students use the Toolkit for CreateJS to output a sample animation to HTML5 and JavaScript. The Toolkit for CreateJS enables a seamless and integrated workflow for both designers and developers.

Toolkit for CreateJS

Toolkit for CreateJS, originally an optional extension for Flash Professional, is a way to publish HTML5 content from Flash. It allows you to leverage Flash's powerful animation and drawing tools while targeting multiple runtimes—both the Flash Player and HTML5. The Toolkit outputs your animation using a suite of open source JavaScript libraries: EaselJS, TweenJS, and SoundJS.

- EaselJS is a library that provides a display list that allows you to work with objects on the canvas in the browser.
- TweenJS is a library that provides the animation features.
- SoundJS is a library that provides functionality to play audio in the browser.

Toolkit for CreateJS generates all the necessary JavaScript code to represent your images, graphics, symbols, animations, and sounds on the Stage. It also outputs dependent assets, such as images and sounds. You can even include simple JavaScript Timeline commands in the Actions panel, which gets exported in the JavaScript files.

However, emphasize to the students that the Toolkit for CreateJS is *not* meant to be a soup-to-nuts exporter to convert all Flash content to HTML5 in a single push of a button. It is not an ActionScript-to-JavaScript translator. It is designed to be part of a larger workflow that offers a simple way to output Flash assets for JavaScript. It creates accessible JavaScript files for a developer to continue to add more complex interactivity.

Classic tweens

While both motion tweens and shape tweens are supported by the Toolkit for CreateJS, it converts those animations to frame-by-frame animations. Classic tweens, an older way of creating animation in Flash Professional, are retained as a runtime tween, which saves file size and allows you to control the animation dynamically via JavaScript. For these reasons, Adobe recommends using classic tweens for animations targeted for output with the Toolkit for CreateJS.

Classic tweening is an older approach to animation, but it is very similar to motion tweening. Like motion tweens, classic tweens use symbol instances. Changes to the symbol instances between two keyframes are interpolated to create the animation.

You can change an instance's position, its rotation, scale, transformation, Color Effect, of Filter effect.

The key differences between classic tweens and motion tweens are:

- Classic tweens require that you begin with a symbol instance on the Stage.
- Classic tweens require a separate Motion Guide layer to animate along a path.
- Classic tweens don't support 3D rotations or translations.
- Classic tweens are not separated on their own tween layer, but classic and motion tweens share the same restriction that no other object can exist in the same layer as the tween.
- Classic tweens are Timeline-based, and not object-based, meaning you add, remove, or swap the tween or the instance on the Timeline and not the Stage.

Inserting JavaScript

The optimal workflow is to use the Toolkit for CreateJS to output animated assets from Flash, and then to have a developer integrate additional interactivity with JavaScript.

However, it is possible to add some JavaScript directly to the Flash Timeline, which gets exported to the published JavaScript files.

In the Actions panel, use /* js to begin a block of JavaScript code, and /* to end the block of JavaScript code. Use JavaScript in the Timeline sparingly; to control the Timeline with the easelJS MovieClip class commands play(), gotoAndStop(), stop(), and gotoAndPlay(), for example.

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. What files are exported when you publish with the Toolkit for CreateJS?
2. How do you know if a particular Flash feature is supported by the Toolkit for CreateJS?
3. How do classic tweens and motion tweens differ in the way they each handle motion on a custom path?

Answers

1. The Toolkit, by default, outputs two files: a JavaScript file that contains all the necessary JavaScript code to represent your images, graphics, symbols, animations, and sounds on the Stage, and an HTML file that displays the animation in the browser. It also outputs any dependent assets, such as images and sounds.

2 When you publish with the Toolkit for CreateJS, Flash warns you of any incompatible Flash features that will not export or work as expected in the Output panel.

3 The path of motion for a motion tween appears in the Tween layer, while the path of motion for a classic tween must be created in a separate Guide layer and paired with the layer containing the classic tween.

Lesson 11: Publishing Flash Documents

Lesson Overview

In Lesson 11, students explore a range of options for testing and publishing a Flash project. They publish a finished banner animation for the Web, they create a desktop application with Adobe AIR, and they test a mobile app within the Flash application.

Understanding publishing

Publishing is the process that creates the required files to play your final Flash project for your viewers. Students need to understand the difference between "authortime" and "runtime." Authortime refers to development within the Flash Pro CC application since it is the environment in which content is being created, or authored. Runtime refers to playback in the targeted viewing environment, such as a desktop browser or a mobile device. Adobe provides various runtime environments to play Flash content. The most common is the Flash Player, which plays Flash content on Web browsers. Adobe AIR is another runtime environment that can play Flash content directly from your desktop, or can play Flash content as an app on mobile devices running Android or the iOS.

Testing a Flash document

One fast way to preview a movie that targets the Flash Player is to choose Control > Test Movie > in Flash Professional (Ctrl+Enter/Command+Return). This command creates a SWF file in the same location as your FLA file so 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 the 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 lesson, 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.

Publishing a movie for the Web

When you target the Flash Player, the Publish command creates a Flash SWF file and an HTML document that inserts your Flash content in a browser window. Flash Player 11.7 is the latest version. If you change publish settings, Flash saves the changes with the document. The default settings for Publishing should be sufficient for most students, but it's useful to point out the various options available for customization.

Publishing an AIR application

You can create a desktop application for Windows or for the Macintosh by targeting Adobe AIR 3.6 for the Desktop in the Publish section of the Properties inspector. An AIR application requires the AIR runtime. You can either download the free AIR runtime from Adobe's site, or you can include the AIR runtime within your application during the publishing process.

To create an AIR desktop application, edit the AIR application settings and provide some additional information:

- You need to provide a certificate so that users can trust and identify the developer of the Flash content. In the Signature section of the Application Settings dialog box, you can guide your students to create their own self-signed certificate, which has the .p12 extension.
- The Icons section of the Applications Settings dialog box allows students to use PNG images as icons for the AIR application.
- The Advanced section of the Applications Setting dialog box contains settings for the initial window location and options for resizing.

Simulating mobile app interactions

Flash provides the SimController, which allows you to simulate mobile app interactions. Since creating an app for mobile devices is a little more complicated because it requires specific developer certificates and the actual hardware, the SimController is very useful for early testing. Interactions such as swiping and pinching, or even tilting the device can be emulated directly within Flash.

In this lesson, students use the SimController to test out mobile interactions in an already completed Flash file that targets AIR 3.6 for Android. To test a mobile app, choose Control > Test Movie > in AIR Debug Launcher (mobile), and the SimController automatically launches.

Mobile interactions that can be tested in the SimController include:

- Using the accelerometer in three directions (x-, y-, and z-axis)
- Click and drag, press and tap, two-finger tap, pan, rotate, zoom, and swipe
- Geolocation
- Menu, Back, and Search buttons

Other testing methods for mobile apps

Flash Professional provides several ways to help testing content for mobile devices, in addition to using the SimController in Test Movie mode.

- For iOS devices, Flash can publish an AIR app to test in the native iOS Simulator, which emulates the mobile app experience on your desktop. The iOS Simulator is part of Apple's XCode developer toolset (https://developer.apple.com/xcode/index.php), available as a free download in the App store.
- Connect a mobile device to your computer with a USB cable and Flash can publish an AIR app directly to your mobile device. (Testing an app via USB on an iOS device requires that you be part of Apple's iOS Developer program, where you create development, distribution, and provisioning certificates. Testing an app via USB on an Android device also required certificates, but you can create your own for testing purposes).

Questions

The following are additional questions that are not in the students' Classroom in a Book.

1. What is the purpose of QA testing?
2. What is Adobe Scout?
3. Which one of these interactions is NOT available in the SimController: geolocation, swipe, pinch, accelerometer, microphone?

Answers

1. QA, or quality assurance testing, is the process where the functionality of a project is tested. QA testing is important to measure the actual performance with the expected performance of any multimedia project.
2. Adobe Scout is an advanced tool for analyzing your Flash content and its performance. Use Scout to profile and optimize any Flash content, whether it runs in a browser or on a mobile device. Scout is a standalone application available through Creative Cloud. Scout gives you a peek behind the scenes of the Flash Player—information such as CPU and memory usage, or how the frame rate of your movie is keeping up or falling behind
3. The microphone function is not available in the SimController.