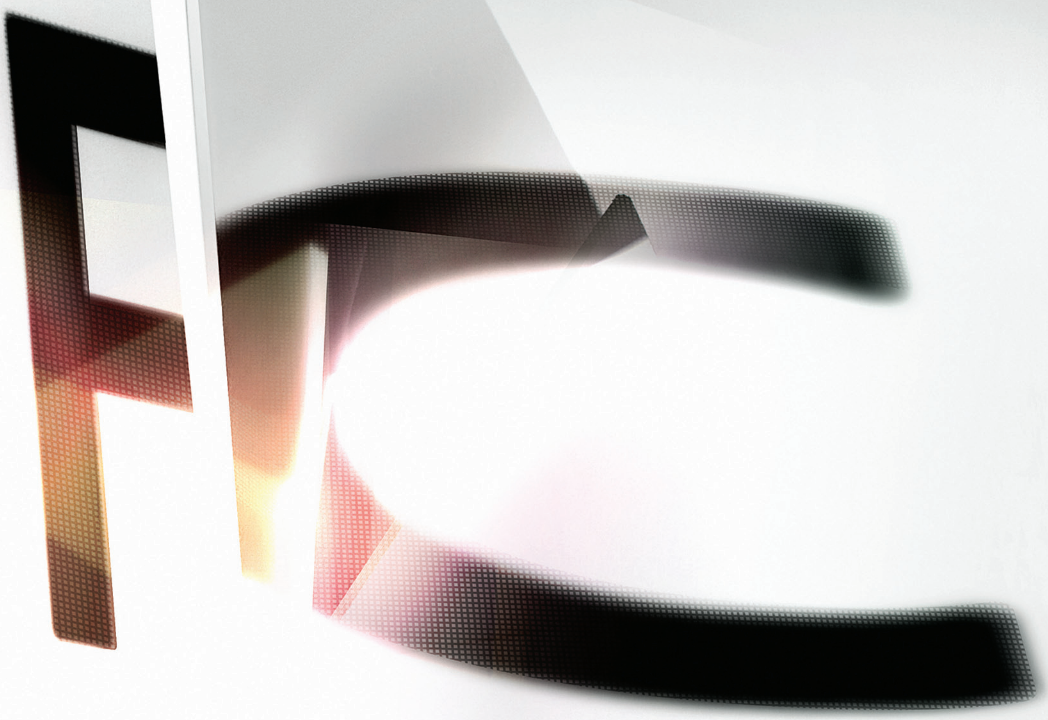
# ADOBE® FLASH® CATALYST™ CS5



# CLASSROOM IN A BOOK®

Instructor Notes

Adobe® Flash Catalyst® CS5 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 Berkeley, 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.

Book:
ISBN-13: 978-0-321-70358-3
ISBN-10: 0-321-70358-8

Instructor Notes:
ISBN-13: 978-0-132-48162-5
ISBN-10: 0-132-48162-6

# INSTRUCTOR NOTES

## Getting Started

The *Adobe Flash Catalyst CS5 Classroom in a Book* course presents students with tips, techniques, and solutions for using the Adobe Flash Catalyst CS5 software. The Instructor Notes are intended to complement the information in the *Adobe Flash Catalyst CS5 Classroom in a Book*. The information is organized to follow the sequence of instruction in each lesson.

### Course strategy

If you're teaching a 15-session class, you can teach approximately one lesson of this book per session. This book contains 13 lessons, some of which may take a while to complete. If you are teaching a 10-session class, you may want to combine some of the lessons in a single class. For example:

- Lessons 1 and 13 both discuss the Flash Catalyst designer/developer workflow, including integration between Flash Catalyst and Adobe Flash Builder. Combined, these lessons will run about 90 minutes.
- Lessons 3 and 4 cover managing the Flash Catalyst Library and Layers panels. These are two of the shorter lessons in the course, and both concepts are interrelated. Combined, these lessons will run about 90 minutes.
- Lesson 5 explains Flash Catalyst pages and states, which is a fundamental concept for working effectively in Flash Catalyst. Although it's not required, some instructors will find it useful to review this lesson before teaching Lessons 2 through 4.
- Lesson 6, Creating Interactive Components, is a longer lesson (90 minutes) and could be taught in two sessions. If you choose to divide this lesson, you can teach the topics on creating components first (about 60 minutes) and teach the final topics on adding interactivity in a separate session (about 30 minutes). The interactivity topics could also be combined with Lesson 7, Creating Transitions and Action Sequences, to create one longer session (about 75 minutes).
- Lesson 8 covers video and sound and Lesson 9 covers SWF files, but you could teach these lessons together in a session on working with multimedia. Combined, these lessons will run about 90 minutes.
- Lesson 12, Publishing a Project, and Lesson 13, Extending Your Project Using Adobe Flash Builder, complement each other and could be taught together. Combined, these lessons will run about 90 minutes.

## Managing student projects

One way to simplify file storage and retrieval in a classroom situation is to ask students to create a folder on their hard disks, name it *[student's name]* Lessons, and then copy each project folder into the main student's lesson folder. Having students keep their working files in their own folders makes it easy for you to clean up the files when class is over.

## New and key features

Flash Catalyst is a brand new program, so every features covered in *Adobe Flash Catalyst CS5 Classroom in a Book* is a new feature. Some key features to reinforce throughout this course include:

- **Pages and States**, introduced in Lesson 1, explained in Lesson 6, and reinforced throughout the course, is one of the most important concepts for students to grasp when using Flash Catalyst.
- **Layers** are introduced in Lesson 2 and covered in greater detail in Lesson 4. It's important that students understand the relationship between layers and states, as well as how the same layers persist across all states in the application. Equally important is that students understand the difference between an object being present, visible, and hidden in a page or component state.
- **Sharing objects to states** is described in Lesson 5 and then reinforced throughout the course. Make sure students understand the difference between adding multiple instances of the same object and sharing the same object to multiple states (and assigning unique properties).
- **Editing components** is taught in Lesson 6 and used throughout the course. Be sure students understand the difference between editing a component using Edit-In-Place mode and applying properties to a component using the Properties panel.
- **Transitions and action sequences** are explained in Lesson 7. An action sequence is technically a type of interaction, and adding interactions is introduced in Lesson 6. Make sure students understand the differences between transitions and action sequences.
- **Scroll Panel and Scrollbar components** are special types of Flex components that cause confusion for some new users. A Scroll Panel component requires you to include a scrollbar and define which content to scroll. A Scrollbar component must always include a track and thumb. These components are explained in Lesson 6.
- **Data lists and design-time data** are powerful features for creating both design and data-centric applications and are covered in Lesson 10.

## Resources

*Adobe Flash Catalyst CS5 Classroom in a Book* is not meant to replace documentation that comes with Flash Catalyst or to be a comprehensive reference for every feature. Only the commands and options used in the lessons are explained in this book. For comprehensive information about program features and tutorials refer to these resources:

**Adobe Flash Catalyst Help and Support:** www.adobe.com/support/Flash Catalyst where you can find and browse Help and Support content on adobe.com.

**Adobe TV:** http://tv.adobe.com is an online video resource for expert instruction and inspiration about Adobe products, including a How To channel to get you started with your product.

**Adobe Design Center:** www.adobe.com/designcenter offers thoughtful articles on design and design issues, a gallery showcasing the work of top-notch designers, tutorials, and more.

**Adobe Developer Connection:** www.adobe.com/devnet is your source for technical articles, code samples, and how-to videos that cover Adobe developer products and technologies.

**Resources for educators:** www.adobe.com/education includes three free curriculums that teach Adobe software through an integrated approach and can be used to prepare for the Adobe Certified Associate exams.

Also check out these useful links:

**Adobe Forums:** http://forums.adobe.com lets you tap into peer-to-peer discussions, questions, and answers on Adobe products.

**Adobe Marketplace & Exchange:** www.adobe.com/cfusion/exchange is a central resource for finding tools, services, extensions, and code samples to supplement and extend your Adobe products.

**Adobe Flash Catalyst CS5 product home page:** www.adobe.com/products/Flash Catalyst

# Lesson 1: Getting to Know Adobe Flash Catalyst

The first lesson presents an introduction to the Adobe Flash Catalyst CS5 software, the Adobe Flash Platform, and the Flex framework for designing and developing RIAs. This lesson explores the features and benefits of Flash Catalyst and how designers are expected to use the program. The lesson includes an example of a designer/developer workflow for creating and publishing Flash Catalyst projects and compares a new blank project and a completed rich media application.

## Goals for this lesson

After completing this lesson, students will be able to:

- Define Flash Catalyst and its target users
- Describe workflows for designing and publishing applications using Flash Catalyst
- Start a new Flash Catalyst project
- Set dimensions and background color
- Navigate the Design and Code workspaces
- Show, hide, and adjust workspace panels
- Open and preview an existing Flash Catalyst project
- Find additional help documents, demos, and tutorials online

## About Flash Catalyst

Before students start building applications in Flash Catalyst they must understand the relationship between Flash Catalyst and the Flex framework. Teaching the Flash platform and Flex is beyond the scope of this course, but you should try to become familiar with these prior to teaching a class on Flash Catalyst.

### Key skills or concepts to reinforce:

- Flash Catalyst is a design tool for creating application interfaces. While creating projects, Flash Catalyst is writing the MXML code and creating the necessary documents for the Flex project.
- Students should be sure to read the sidebars "The Adobe Flash Platform" and "Adobe Flex framework and MXML."

## Flash Catalyst workflow

The workflow and illustration in Lesson 1 shows a design-centric workflow for an application created and published using Flash Catalyst. Students may ask about the absence of Flash Builder in the workflow illustration. Refer to Lesson 13 for an alternate version of the workflow and illustration for projects that require extended development using Flash Builder.

### Additional points for discussion or clarification:

The workflow example and illustrations used in this course are not the only possible workflows. The steps shown in the Lesson 1 workflow are flexible. For example, some designers will prefer to convert artwork into functional components before adding new page states. Other designers may want to add transitions and action sequences before designing smooth transitions. When discussing possible workflows, you can mention that there are really several possible workflows. A few possible workflows include:

- **Interactive content with no dynamic data.** Originate artwork in Creative Suite documents, complete interaction design in Flash Catalyst, and publish from Flash Catalyst.
- **Linear design and development with no iteration.** Begin with Creative Suite, move into Flash Catalyst, and complete the development in Flash Builder.
- **Iterative development (single project/small team).** Some iteration between Flash Catalyst and Creative Suite. Small projects allow for some changes to the Flash Catalyst project file and merging changes in Flash Builder (see Lesson 13 for more information).
- **Prototype-lead development.** Use the wireframe components in Flash Catalyst to create the scaffold of the application, that is a version of the application that contains placeholder components, with states and interactions defined for the basic interaction the application but without the visual design applied. With this approach, the scaffold project can be opened in Flash Builder and application development work can continue while waiting for the design artwork to be completed and approved.
- **Iterative development (multi-project/large team).** Design in Flash Catalyst and export a library package (FXPL). An FXPL contains only the component skins, item renderers, custom components, and supporting assets that you have defined in your project and not the main application MXML document. This makes the Flash Catalyst Library Package ideal for sharing a set of re-useable component designs across multiple Flash Catalyst projects and enables you to deliver updated sets of component skins in a single package for use in Flash Builder. See Lessons 3 and 13 for more information on exporting and importing FXPL files.

For more information on these and other possible workflows, view *Exploring Flash Catalyst and Flash Builder Workflows*, by Andrew Shorten, on the Adobe Developer Connection (www.adobe.com/devnet/flashcatalyst/articles/flashbuilder_flashcatalyst_workflows.html).

Some designers might disagree with the definition of microsite that was used in the discussion on Flash Catalyst workflows. That's OK. The important message here is that designers know they can design and publish a completely functional and final application from within Flash Catalyst without additional development using Flash Builder. On the other hand, some applications will require support from a developer to bind user interface controls to data and services. You may also have design-centric projects that require binding to data.

## Opening a new Flash Catalyst project

Students open a new blank project from the Flash Catalyst welcome screen. When opening a new blank project, students may notice the absence of the option to create a new project directly from Adobe Fireworks.

### Additional points for discussion or clarification:

- Explain that artwork created in Fireworks must be exported from Fireworks as an FXG file. You can then click From FXG File in the Welcome screen to create a new project from the Fireworks file.
- Explain that all artwork you import into Flash Catalyst is converted to FXG format during the import process. The FXG file format and how to export FXG files are covered in more detail in Lesson 2.

## Finding your way around

Students explore the Flash Catalyst user interface, including the Design and Code workspaces.

### Additional points for discussion or clarification:

- This is the first time students will see the Heads-Up Display (HUD). Throughout this course, student will use the HUD to create and modify components, optimize graphics, and more. Make sure students know how to show, hide, and move (by dragging) the HUD, as it will need to be moved frequently as they work.
- Let students know that any commands found in the HUD can also be found in the Flash Catalyst menus at the top of the window. Anytime teaching a skill that utilizes the HUD, remember to also point out the menu equivalent.
- The Breadcrumbs bar is a point of confusion for some students. To help them understand, consider providing a preliminary overview of working with components and Edit-In-Place. Be careful not to get too far off track, though. There is plenty of time to discuss and demonstrate these features later in the course.
- Demonstrate how to locate and select tools that share the same space in the Tools panel (by clicking and holding the tool). While you're at it, point out the keyboard shortcuts that appear for some (but not all) tools.
- When exploring the Code workspace, students may want to know why they can view the MXML code for a project in the Code workspace, but they cannot edit the code. Let them know this is a 1.0 release of Flash Catalyst and the ability to edit code in Flash Catalyst is something that is most likely coming in the future. Code view helps students understand what's going on behind the scenes and identify potential issues with the project. When an issue appears in

the Problems panel, you can double click the problem to locate the error in the code. Sometimes by identifying issues in the code, you can resolve problem by making change to the project in Design view.

- For some students, it's confusing that their Flash Catalyst project includes a collection of files in the Project Navigator (Code workspace). This confusion is caused by the fact that they have to open and edit only one file, the FXP project file. When they share a project file with another designer, they share a single FXP file only, not a directory of files. One way to explain this is to compare the FXP file to a ZIP file that stores the entire collection of project files and structure. These files and the file structure become more apparent when you publish the project for the web or when importing the project into Flash Builder for further development.

## Opening an existing Flash Catalyst project

Students open and run an existing project called RIA.fxp. You can draw attention to the fact that this project has only three main pages listed in the Pages/States panel, but there are several different views and lots of depth. This result is accomplished through nested components, which are a great way to improve the performance of a project. As students will learn in Lessons 5 and 6, a project (or custom component) can have no more than 20 states. Having too many main application page states can slow performance. A best practice is to use custom components and nested components to create multiple views or states within a single page of the application. The RIA.fxp project is a good example of this.

## Previewing a project in a browser

Students preview the RIA.fxp project in a web browser by choosing File > Run Project. The file opens in the student's default web browser. Unlike some other web design applications, such as Adobe Dreamweaver, in Flash Catalyst you cannot select from a list of browsers when running the project. If you want to test the project in another browser, you need to either change your default browser or publish an offline version of the application and then open the main.html file in each browser. Publishing a project is covered in Lesson 12.

## Getting Flash Catalyst help

Students choose Help > Flash Catalyst Help and use the Search box to locate online support for working with Flash Catalyst.

### Additional points for discussion or clarification:

- Sometimes students are confused about the Search bar's functionality. The Search bar is used to access online help, not search for content in the Flash Catalyst project.

- When accessing Adobe Community Help, students may be required to update Adobe AIR. Follow the prompts to complete the update. Students may also be required to update their Local Help Content. If prompted to update local help, choose Update.
- The content located in the Adobe Community Help system is dynamic, so what students see one week may differ from what they see another week. Encourage students to visit the Community Help system often for current tips and ideas on using Flash Catalyst.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1 What is the Adobe Flex framework?

2 There aren't many drawing or graphic editing tools available in the Tools panel. How can you perform more advanced graphics editing in Flash Catalyst?

3 How can you find the keyboard equivalent for a tool in the tools panel?

4 How can you identify errors in the MXML code of a Flash Catalyst project while working in Flash Catalyst?

5 How can you limit the number of main application pages while presenting a wide range of content using multiple screens or views?

6 What happens as a result of creating a new Flash Catalyst project from an Illustrator file with multiple artboards, as compared to creating a project from an Illustrator file with a single artboard?

## Answers

1 The applications you build with Flash Catalyst are Flex applications. Flex is an open source framework for building and deploying applications that run on all major web browsers, desktops, and operating systems. MXML is the language developers use to define the layout, appearance, and behaviors in Flex applications. ActionScript 3.0 is the language used to define the client-side application logic. When you publish a Flash Catalyst project, your MXML and ActionScript are compiled as a SWF file.

2 Flash Catalyst includes a basic set of drawing tools, but for more advanced drawing and editing, you can use the integrated launch and edit features for round-trip editing between Flash Catalyst and Adobe Photoshop or Illustrator. You will learn more about round-trip editing in Lesson 11.

3 Hold the pointer over the tool until the tool tip appears showing the keyboard equivalent for the tool. Not all Flash Catalyst tools in the Tools panel have keyboard equivalents.

4. To identify code issues, open the Code workspace and view the Problems panel. You can double-click any issues to locate the location of the error in the code. You cannot edit MXML code in Flash Catalyst, but you may be able to resolve the problem by making changes to the project in the Design view, or by bringing the errors to the attention of a Flex developer.
5. Flash Catalyst lets you create custom components with up to 20 states. Each of these components can include other (nested) components, and you can place multiple components in a page. Creating custom components and nesting components is a good way to increase an application's depth of content without exceeding the 20-page limit. Having too many main application pages can reduce the overall performance of the application.
6. When you import an Illustrator file with a single artboard, all artwork is placed on one new page state in Flash Catalyst. When you import an Illustrator file with multiple artboards, the contents of each artboard are placed on separate page states in Flash Catalyst.

## Lesson 2: Preparing, Importing, and Placing Artwork

There are really two parts to this lesson. First, students will learn tips and best practices for preparing a design document for Flash Catalyst. You can design the artwork for a project using Adobe Illustrator, Adobe Photoshop, or Adobe Fireworks, but Flash Catalyst has better product integration with Illustrator and Photoshop. The processes for importing documents created in Illustrator, Photoshop, and Fireworks are all a little different. In an attempt to avoid too much redundancy, *Adobe Flash Catalyst CS5 Classroom in a Book* combines many of the common steps and calls out the differences where they exist, either in sidebars or in tips and notes.

In the second part of this lesson, students import a design that was created in Illustrator. Students will also import and place additional images to see how that differs from importing a structured design document.

### Goals for this lesson

After completing this lesson, students will be able to:

- Prepare a design document for import
- Export an FXG file from Adobe Fireworks
- Import a layered design document into Flash Catalyst
- Select fidelity options when importing a design document

- Import artwork into Flash Catalyst
- Position artwork in the Flash Catalyst artboard
- Optimize artwork in Flash Catalyst

## Preparing a design document

This topic includes tips and ideas for preparing a design document that you will import into a Flash Catalyst project. A *design document* is a structured/layered collection of artwork for an application interface that you've created in Adobe Illustrator, Photoshop, or Fireworks.

### Additional points for discussion or clarification:

- **Choosing your design application:** By Lesson 2, students are aware they can start a new Flash Catalyst project by opening a document created in Illustrator, Photoshop, or Fireworks. They may not be sure why they would choose one program over another. Let students know they can work in the application in which they are most experienced, but stress that Flash Catalyst imports files in the FXG file format. Make sure they read the sidebar on FXG. Because of the integration between Flash Catalyst and Illustrator and Photoshop, these are the preferred applications for creating layered designs. When importing Illustrator and Photoshop documents, the files are converted to FXG automatically. To import artwork created in other programs, you first need to convert the file to FXG and then import the FXG file into Flash Catalyst. Point out that other programs, such as Adobe In Design, can also output FXG documents.
- **Flash Catalyst document profile:** A tip in Lesson 2, recommends using the Illustrator Flash Catalyst document profile/preset when creating a design document for Flash Catalyst. This automatically sets the color mode to RGB, the PPI to 72, and enables the Align To Pixel Grid option. The Flash Catalyst document profile is available in Illustrator CS5. The Flash Catalyst document profile is available in the Illustrator Welcome Screen and by choosing File > New and opening the New Document Profile pop-up menu and choosing Flash Catalyst.
- **Objects with filters and effects:** Flash Catalyst users may be unsure when to flatten layers prior to import and when to rely on the Flash Catalyst import fidelity options. In general, Flash Catalyst does an excellent job at determining which attributes to leave editable and which to flatten. When in doubt, use the default import fidelity options. Flash Catalyst will display a message if your document includes items that require further optimization.
- **Structuring applications:** In *Adobe Flash Catalyst CS5 Classroom in a Book*, students import an Illustrator file with a single artboard and begin with all artwork in a single page state. They will use the Pages/States and Layers panels (later in this course) to create the page structure for the application. Using this

approach, artwork is automatically shared between states. Sharing objects between states makes for better transitions and consistent placement of shared objects. Some designers will prefer to begin defining page structure using multiple artboards (Illustrator) or Layer Comps (Photoshop). Photoshop Layer Comp (composition) provides a snapshot of which layers are turned on and which layers are turned off in your Photoshop document. When preparing a document for import to Flash Catalyst, you can use Layer Comps to represent each page state in your Flash Catalyst application. You can then import each Layer Comp to a different state in Flash Catalyst. Let students know that both approaches (single artboard or multiple artboards/Layer Comps) are fine, but the method they choose will affect how they work with artwork in Flash Catalyst.

## Importing artwork (and import fidelity)

In this topic, students import a layered design document created in Adobe Illustrator. Import fidelity options are explained in a sidebar. Students may have additional questions about the differences between fidelity options when importing Photoshop and Illustrator files. You can refer to the following descriptions when answering questions or explaining the subtle differences between Photoshop and Illustrator import fidelity.

### Illustrator Import Fidelity Options:

- **Filters** > **Keep Editable** preserves editable Illustrator filters that are supported in Flash Catalyst. Supported filters include Blur (Gaussian Blur), Drop Shadow, Inner Shadow, Bevel, Glow, Inner Glow. After import, you can edit filters in the Properties panel.
- **Filters** > **Expand** converts objects with filters into sets of vectors or bitmaps that approximate the original appearance in Illustrator. For example, expanding a drop shadow on a rectangle creates an image of the drop shadow behind the original rectangle. The image and the shadow are separate objects within a group in the Layers panel.
- **Filters** > **Flatten** converts objects with filters to single bitmap (raster) images. The image and filter are one non-editable image.
- **Text** > **Automatic Conversion** lets Flash Catalyst choose when to keep text editable, rasterize it, or convert it to vectors. The result is based on various heuristics, such as the complexity of the item. For example, a simple blend expands to vectors, but a complex blend, which produces lots of paths, is rasterized.
- **Text** > **Keep Editable** imports Illustrators text as editable text objects. Once you import editable text, you can change it, size the text bounding box, and apply text formatting properties in the Properties panel. Editable text can lose some fidelity on import.

- **Text > Vector Outlines** converts text into vector paths that approximate the rendered appearance of the text.
- **Text > Flatten** converts text into a single bitmap (raster) image.
- **Gradients > Automatic Conversion** lets Flash Catalyst choose when to keep gradients editable or to rasterize.
- **Gradients > Keep Editable** keeps objects with Illustrator gradients editable. Flash Catalyst supports linear gradients only. Linear gradients are editable in the Properties panel. Objects with radial or elliptical gradients remain editable (not rasterized), but their gradients cannot be edited in Flash Catalyst. You can change the fill to Solid, Gradient (linear), or None in the Properties panel.
- **Blend > Automatic Conversion** lets Flash Catalyst choose whether to keep objects with blends editable or to rasterize them. The result is based on such heuristics as the complexity of the item and supported blends.
- **Flatten** converts objects with blends into a single bitmap (raster) image.

### Photoshop Import Fidelity Options:

- **Image Layers > Keep Editable** keeps image layers with effects editable. Layer effects import as a separate objects in the Flash Catalyst Layers panel.
- **Image Layers > Flatten** converts image layers with effects into a single bitmap (raster) image.
- **Shape Layers > Crop** imports the vector mask that defines a shape but crops the resulting bitmap to the bounds of the mask.
- **Shape Layers > Flatten** converts shape layers with effects into a single bitmap (raster) image.
- **Text Layers > Keep Editable** imports text layers as editable text objects. Once you import editable text, you can change it, resize the text bounding box, and apply text formatting properties in the Properties panel. Editable text can lose some fidelity on import.
- **Text Layers > Vector Outlines** converts text into vector paths that approximate the rendered appearance of the text.
- **Text Layers > Flatten** converts text layers with effects into a single bitmap (raster) image.

## Optimizing artwork

This topic defines the graphic optimization options that are available from the Flash Catalyst Heads-Up Display and in the Modify menu.

### Key skills or concepts to reinforce:

- Vector artwork is not stored in the project library. If you optimize a vector drawing, however, it's replaced by a new Flash Catalyst Optimized Graphic file and stored in the project library.
- Optimizing large images or vectors with lots of paths can help increase the performance of an application at runtime.

### Additional points for discussion or clarification:

It's always a good idea to optimize graphics before import into Flash Catalyst. There are several ways to optimize the performance of your application before bringing artwork into the project.

- Make sure that bitmap/raster images are 72 PPI.
- Make sure that the color mode is RGB.
- Use Align To Pixel Grid in Illustrator.
- When designing artwork for Flash Catalyst using Adobe Illustrator, you can use the Flash Catalyst document profile to automatically apply optimum settings. The profile applies settings for document size, color mode, PPI, and Align To Pixel Grid.
- When creating videos, use a preset or codec and compression settings that are optimized for the web.

One of the best ways that you can optimize your application is to make the efficient use of assets, pages, states, and components. When building an optimized application, try to:

- Optimize graphics.
- Use a single instance of an asset and share it to states. Students will learn more about this in Lessons 5 and 6.
- When adding multiple videos to a project, use one instance of a video player. You can edit its properties in different states to display and control a different source video. Students work with video in Lesson 8.
- Limit the number of main application pages in your application. To create more views or screens, use the different states of a custom component. Use nested components to create sites with more depth of content and fewer pages. Students will learn more about nesting components in Lessons 6, 8, and 10.

## Working with Photoshop Layer Comps

In *Adobe Flash Catalyst CS5 Classroom in a Book*, Photoshop Layer Comps and the advanced Photoshop import options are introduced, but students are not required to use the advanced import options or Layer Comps when completing the course projects. If you would like to show students how to choose advanced Photoshop import options or work with Layer Comps, you can use the following steps to prepare and deliver your demonstration.

### Preparing a sample Photoshop document with Layer Comps:

1. Create a new Photoshop document with at least two layers. Add some content, such as images, text, or shapes in each layer. Make sure each layer's contents are noticeably different. Each layer represents a different page in your sample application.
2. In the Photoshop Layers panel, name one layer Page 1 and name the other layer Page 2.
3. In Photoshop, choose Window > Layer Comps. In the Photoshop Layers panel, hide the layer for Page 1. In the Layer Comps window, click the Create New Layer Comp icon, name the Layer Comp Page 1, and click OK. Create a second Layer Comp named Page 2 that shows only the contents in the Page 2 layer.
4. Save the Photoshop file and name it Sample Layer Comps.psd.

### Importing Photoshop Layer Comps using the advanced import options:

1. Start Flash Catalyst. In the Create New Project from Design File section of the Welcome screen, click From Adobe Photoshop.psd File, locate the file Sample Layer Comps.psd, and click Open.
2. Click Advanced in the Photoshop Import Options dialog box.

   You should mention to students that the name of the advanced dialog box differs between Windows and Mac OS versions of Flash Catalyst. In Windows it's called Import [*document name*] To State. In Mac OS it's called Select Photoshop Layer Comp.
3. Because you're opening a Photoshop document with Layer Comps, a pop-up menu called Select Photoshop Layer Comp is available This menu appears only if your PSD file includes Layer Comps. Open the Select Photoshop Layer Comp menu, and choose Page 1. Notice that the Page 2 layer is not selected for import.

   You can also choose Last Document State. This imports artwork in any layers that were visible when you saved the Photoshop file.
4. In the Check Photoshop Layers to Import section of the dialog box, click Page 1. Notice that fidelity options for this layer are shown to the right. Explain that you can always use the advanced Photoshop import options (with or without Layer Comps) to select which layers to import and then set fidelity options for each layer individually. This is not an option when importing an Illustrator document.
5. With Page 1 selected as the Layer Comp to import, click OK. The Page 1 Layer Comp is added to the Page1 state in the new Flash Catalyst project.

6 In Flash Catalyst, click New Blank State in the Pages/State panel. A new blank page (Page2) is added. Choose File > Import > Adobe Photoshop File (.psd). Select the file Sample Layer Comps.psd, and click Open. Click Advanced in the Photoshop Import Options dialog box. Select the Page 2 Layer Comp from the pop-up menu, and click OK. The second layer comp is added to the second page in the Flash Catalyst document.

When you import artwork in a Photoshop Layer Comp, the artwork is centered in the Flash Catalyst artboard. To preserve the layout and positioning of your Layer Comps, include a background layer that is the same size as the Flash Catalyst artboard.

## Importing Illustrator symbols

Some students are confused about the relationship between Illustrator symbols and Flash Catalyst components. In Illustrator, symbols are reusable objects stored in a library. When you edit the symbol, you edit every instance of the symbol in the Illustrator document. Flash Catalyst components are similar, but when you import an Illustrator symbol, they are not converted to Flash Catalyst components automatically. Instead, Illustrator symbols import as Flash Catalyst Optimized Graphics. If your Illustrator file includes multiple instances of the same symbol, then your Flash Catalyst document stores one new Optimized Graphic in the Library panel and places multiple instances of the same optimized graphic in your project. In Flash Catalyst, it is a best practice to use one instance of an object and then share that object to other states. After importing the Illustrator file, you should remove all but one instance of the optimized graphic. Then, share the same instance of the optimized graphic to other states, and then apply different properties in each state, as needed. To convert the optimized graphic into a Flash Catalyst component (for example if you want to turn the artwork into a button), you must first break it apart by choosing Modify > Break Apart Graphic.

Remind students that they can always see the source of an object by selecting the object (in the artboard or Layers panel), and then looking at the Source link in the Properties panel.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1 Which settings are applied automatically when using the Adobe Illustrator CS5 Flash Catalyst document preset?

2 Which additional import options are available by clicking Advanced in the Photoshop Import Options dialog box?

3 After creating a new Flash Catalyst project, how do you import additional design comps (into your existing project) from another Illustrator, Photoshop, or FXG file?

4 When creating sample artwork for the records in a Data List component, why would you group the artwork for the first record and create a separate group for the remaining sample records?

5 What happens to Adobe Illustrator symbols when importing an Illustrator file into Flash Catalyst?

6 How can you edit the individual parts of a vector drawing that has been converted to an Optimized Graphic in Flash Catalyst?

## Answers

1 The Illustrator Flash Catalyst document profile/preset automatically sets the color mode to RGB, the PPI to 72, and enables the Align To Pixel Grid option.

2 When you click Advanced in the Photoshop Import Options dialog box, you have the option of selecting which layers to import. You can also set import fidelity options for each layer individually. If the Photoshop document includes Layer Comps, you can choose a Layer Comp to import.

3 With your Flash Catalyst project open, you can choose File > Import and select the type of file to import (.psd, .ai, or .fxg). The artwork you import is added to the current state in the existing project. Any bitmaps in the new file are stored in a separate new folder in the Images category of the Library panel.

4 When designing artwork for a Data List component, you don't need to include every item in the list, just a representative model. A best practice when preparing your design document is to include the first item and a separate group that serves as a guide for alignment and spacing. You use the first item in the list as the repeating part of the Data List component. The second group is used for spacing and alignment only and can be thrown away and replaced with design-time data when you create the Data List component. Lesson 10 covers creating a Data List component.

5 Illustrator symbols import as Optimized Graphics. If your Illustrator file includes multiple instances of the same symbol, then your Flash Catalyst document will include multiple instances of the same optimized graphic.

6 To edit the parts of an optimized graphic, you must first break it apart. Select the optimized graphic. In the Optimize Artwork section of the HUD, click Choose Command, and select Break Apart Graphic. Or select the graphic and choose Modify > Break Apart Graphic in the menus.

# Lesson 3: Managing the Library

In this lesson, students explore the Flash Catalyst project library. Students gain a clear understanding of the difference between projects and the application they create within a project. They will also be able to explain the difference between project library assets and the objects they include in the application.

## Goals for this lesson

After completing this lesson, students will be able to:

- Explain how assets get into the project library
- Add new assets to the project
- Identify project assets in the Library panel
- Preview images, video, and sound
- Rename assets in the library
- Delete assets from the project
- Use the library to add assets to the application
- Use the library to swap assets in the application
- Compress images in the library
- View the source of library files in the Code workspace
- Import and export a library package

## Viewing assets in the Library panel

In this topic, students learn about the types of assets that are stored in the library and how assets get into the library. They add additional assets to the library and preview them using the Library panel.

### Key skills or concepts to reinforce:

- The Library panel does not include vector drawings (that you import or draw), including Flash Catalyst text.
- You cannot import vector drawings by using the File > Import > Image command as you do with bitmap images. You can only import bitmap images in the PNG, GIF, JPG, JPE, and JPEG file types. To import vector drawings, import the Illustrator file or copy and paste the artwork. When you import or copy and paste drawings from Illustrator, the Illustrator import fidelity options appear and the artwork is converted to the FXG file format during import.

## Deleting Library assets

In this topic, students delete library assets and see how this affects content in the artboard. Managing and deleting objects in the artboard and assets in the library is a point of confusion for some new Flash Catalyst users, but it's really very easy once you understand a few basic principles. These are reinforced throughout *Adobe Flash Catalyst CS5 Classroom in a Book*, but you'll be better equipped to support students and answer their questions if you review the following concepts.

### Key skills or concepts to reinforce:

- A Flash Catalyst project includes the application that you're building and the project library, which includes a collection of assets that you can use and reuse throughout the application.
- Assets (components, images, media, and optimized graphics) are stored in the library. They can exist in the library, even if they are not being used in the application.
- Your application can include instances of library assets as well as vector artwork that you import or draw in Flash Catalyst. The vectors, however, are not stored in the project library. To store vectors (drawings or text) in the project library, you must be convert them to components or optimize them.
- You can add one or more instances of a library asset to the application.
- You can share the same artwork or the same instance of a library asset to multiple states in the application. These objects can have different properties in each state. A best practice is to use one instance of an asset in the application or in a component, and then share that same asset across multiple states.
- When you add an object (vector or library asset) to the application, it appears in the current state of the artboard and in its own row in the Layers panel.
- If you select an object in the application (by clicking it in the artboard or in the Layers panel) and press Delete, you remove the object from the current state only.
- If you select an object (by clicking it in the artboard or in the Layers panel) and then click the Delete icon (trash can) in the Layers panel, you remove the object from every state of the application. If the object is an asset in the library, it still remains in the library.
- If you select an asset in the library and press Delete nothing happens (as long as nothing was selected in the artboard). If you select an asset in the library and click the Delete icon (trash can) in the Library panel, you remove the library assets from the entire project, including all instances in all states of the application.

## Compressing images in the library

In this topic, students compress an image in the library and see how changing a library asset affects all instances of that asset in the application.

**Key skills or concepts to reinforce:**

- When you optimize a vector, it becomes a new optimized graphic in the library.
- When you optimize a bitmap, you are optimizing an image that is already an asset in the library. This creates an optimized copy of the image, and the original bitmap remains in the library.
- If you select an image in the artboard and optimize it, the instance of the image is replaced by the new optimized graphic. If you've already shared the image to other states, the new optimized graphic exists in those states as well. However, if your image had unique properties applied in different states, most of those properties are lost. X and Y position values are preserved, but all other properties, such as size, opacity, rotation, and filters are lost.

## Renaming project assets

Students learn to rename assets stored in the project library.

**Key skills or concepts to reinforce:**

- Renaming assets in the library does not change the names you've used for instances of the asset in the application. In other words, if you rename an asset in the Library panel, it does not change the names you've assigned in the Layers panel.
- Assigning short, descriptive, and unique names to assets in the Library panel is important because these same names appear in the underlying MXML code. Naming library assets makes your project assets easier to work with when shared with another designer or developer.

## Using assets in the Library panel

To add an asset to the application, students learn to drag assets from the Library panel into the artboard.

**Key skills or concepts to reinforce:**

- Avoid using multiple instances of the same asset in the application, unless you need that asset to appear within different components. If you want the same asset to appear on multiple pages, or in multiple states of the same component, share the component to other states.
- If you copy and paste a bitmap, the image is added in the Layers panel and a new asset is placed in the project library.

- If you copy and paste a vector graphic into the artboard, a new object is added to the Layers panel but not the Library panel. That's because vector drawings are not stored in the library until you optimize them or convert them to a component.

## Sharing project libraries

Flash Catalyst project libraries can be exported to share with a developer using Flash Builder or imported into another Flash Catalyst project. This topic provides the steps for exporting and importing a Flash Catalyst Library Package.

### Key skills or concepts to reinforce:

Vector artwork, including the artwork you import and draw, is not stored in the project library. When you export a library package (FXPL) for sharing with a developer or for use in another project, the vectors are not included. To include your vector drawings, convert them to optimized graphics or components before sharing them across other states and adding transitions or interactivity. Converting your artwork to components or optimized graphics adds them to the library, so they'll be available as part of the library package.

## Questions

The following questions are not covered in the student's *Adobe Flash Catalyst CS5 Classroom in a Book.*

1. What happens to the name of an image in the artboard/Layers panel if you change the name of its source asset in the Library panel?
2. What bitmap images (file types) can you import directly into a Flash Catalyst project library?
3. When you compress or optimize an image asset in the Library panel, what happens to the original image asset in the library?
4. What happens if you optimize an image in the artboard that has already been shared to multiple states and the asset has unique properties in each state?
5. What happens if you select an asset in the Library panel and then press the Delete key?
6. Why is it important to give short, descriptive, unique names to assets in the Library panel?

## Answers

1. Nothing. Layer names and library asset names are not linked.
2. PNG, GIF, JPG, JPE, and JPEG.

3 The original uncompressed image remains in the library.

4 If you select an image in the artboard and optimize it, the instance of the image is replaced by the new optimized graphic. If you've already shared the image to other states, the new optimized graphic exists in those states as well. However, if your image had unique properties applied in different states, most of those properties are lost. X and Y position values are preserved, but all other properties, such as size, opacity, rotation, and filters are lost.

5 If you select an asset in the library and press Delete, nothing happens (as long as nothing was selected in the artboard). If you select an asset in the library and click the Delete icon (trash can) in the Library panel, you remove the library asset from the entire project, including all instances in all states of the application.

6 Assigning short, descriptive, and unique names to assets in the Library panel is important because these same names appear in the underlying MXML code. Naming library assets makes your project assets easier to work with when shared with another designer or developer.

# Lesson 4: Managing Layers

The Layers panel (combined with the Pages/States panel) is the primary tool for creating and maintaining the structure of your application. In this lesson, students use the Layers panel to add, remove, show, hide, and arrange (stack) content in the artboard.

## Goals for this lesson

After completing this lesson, students will be able to:

- Identify target and selected layers
- Expand and collapse layers
- Show and hide objects in the current page or state
- Lock and unlock layers, groups, and objects
- Group complex object parts
- Optimize complex groups
- Add and delete layers
- Rename objects in the Layers panel
- Use layers to locate selected objects
- Change the stacking order of objects

## Exploring the Layers panel

In this topic, students explore the layout and features in the Layers panel. It can be difficult for students to understand the Layers panel before being introduced to the concept of pages and states, but students also need to know how to manage layers before they begin to create and modify new pages and states. As you discuss the Layers panel, you will find it helpful to begin explaining pages and states. Pages and states are discussed in greater detail in Lesson 5.

### Key skills or concepts to reinforce:

- The Layers panel persists across all states in the application.
- When you import a design document created in Adobe Illustrator, Photoshop, or Fireworks, Flash Catalyst maintains the integrity of the original design. With your artwork organized into layers, you can begin creating the different pages, components, and component states for the application.

The following summarizes how Flash Catalyst layer terminology compares to layer terminology in Illustrator, Photoshop, and Fireworks.

- A Flash Catalyst layer is the same as an Illustrator layer, a Photoshop layer group, and a Fireworks layer folder.
- A Flash Catalyst object is the same as an Illustrator object, a Photoshop layer, and a Fireworks layer.

### Additional points for discussion or clarification:

- Sometimes people refer to all rows in the Flash Catalyst Layers panel as layers. *Adobe Flash Catalyst CS5 Classroom in a Book* uses the terms “layer” and “sublayer” to refer to the top-level layers identified by the layer folder icons. Flash Catalyst layers and sublayers contain objects (images, text, drawings, component, video, and so on) and groups (grouped objects). Objects exist in stacked rows in the Layers panel.
- Throughout this course students will select objects in the artboard and in the Layers panel. Both methods do the same thing—select objects in the artboard. Selecting artwork using the Layers panel has its advantages. You can be very specific about what you’re selecting without accidentally selecting other objects that are stacked above or below the target object. Some objects can be difficult to select in the artboard, such as transparent objects or objects that are hidden behind other objects. The advantage of selecting objects directly in the artboard is that you can see what you’re selecting (not just relying on a name).
- Selecting a contiguous or non-contiguous range of objects in the Layers panel is different than selecting multiple objects in the artboard. In the artboard, you can Shift-click to select multiple objects. In the Layers panel Shift-click selects a contiguous range (rows) of objects, but Control-click (Windows)

or Command-click (Mac OS) selects non-contiguous rows. Control-click/Command-click does not work to select multiple objects in the artboard.

- Use the Layer panel to control the presence and visibility of objects in each state. Make sure students understand the definitions provided for presence and visibility. This is a prerequisite to working with the Layers panel and much of what they'll be asked to do throughout this course.

## Grouping objects

In this topic, students explore grouped objects in the Layers panel. It's a good idea to group related artwork before importing. Flash Catalyst preserves the groups you create in Photoshop and Illustrator. Each object in the group appears in a separate row in the Layers panel, making them easy to work with.

## Renaming objects in the Layers panel

Students learn to rename objects in the Layers panel. This is something they will do often throughout the course.

### Key skills or concepts to reinforce:

- When you import artwork from a design document created in Adobe Illustrator, Adobe Photoshop, or Adobe Fireworks, the objects in the Layers panel are named automatically using the original layer names you provided in the design document.
- When you create a new group in Flash Catalyst (by selecting artwork and choosing Modify > Group), the new group is given the generic name Group. Be sure to rename the group so you don't end up with several rows in the Layers panel with the same name. The same is also true when creating new components.
- The names you assign to layer folders, groups, and individual objects in the Layer panel are used for organizational purposes only. Changing an object's name in the Layers panel does not affect its name in the Library panel, and vice versa.

## Optimizing complex groups

In this topic, students learn how to turn a group with lots of smaller paths into one optimized graphic. The optimized graphic is easier to manage and is included as an asset in the project library. For example, when you convert text to vector outlines in Illustrator or during the import process, the single text object is turned into a group with multiple smaller paths (one for each character). In most cases, you'll want to optimize the group to create a single object.

## Adding and deleting layers

In this topic, students learn to add and delete new layers and sublayers.

### Key skills or concepts to reinforce:

- Pressing the Delete key removes selected objects from the current state only. If these objects are present in other states, they appear dimmed in the current state, but the row still exists in the Layers panel.
- To remove an object from the Layers panel, select it and click the Delete icon (trash can) in the Layers panel (do not press the Delete key).
- Flash Catalyst includes four Delete icons (trash cans) that all look identical. Make sure that students are using the correct icon when deleting layers and objects in the Layers panel.

## Stacking artwork using layers

In this topic, students learn to change the stacking order of artwork using the Layers panel.

### Key skills or concepts to reinforce:

- The same layers persist across all states in the application, but you can change the stacking order of these objects. However, you cannot have a different stacking order in each page or component state.
- To change the stacking order of objects within the same layer or group, you can drag rows in the Layers panel or use the Arrange commands in the Modify menu.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1. How do you select a non-contiguous range of objects (rows) in the Layers panel?
2. What is the advantage of selecting objects using the Layers panel instead of selecting them directly in the artboard?
3. How can you quickly tell if an object is present somewhere in the current page or component state?
4. How can you remove an object from all states in the application, without removing any assets from the project library?
5. What happens to Illustrator text that has been outlined (turned into vector outlines) when it's imported into Flash Catalyst?
6. How do you change the stacking order of objects within the same layer or group?

## Answers

1. Control-click/Command-click to select a non-contiguous range of objects (rows) in the Layers panel.
2. Selecting objects in the Layers panel helps you to isolate exactly what you want, without accidentally selecting other objects in the artboard. Some objects can be difficult to select in the artboard, such as transparent objects or objects that are hidden behind other objects.
3. When an object exists in the current state, its name in the Layers panel appears dark. When an object exists in the application (in another state), but is not present in the current state, its name appears dimmed in the Layers panel.
4. Select the object in the Layers panel, and click the Delete icon (trash can) in the Layers panel.
5. The text imports as a group of vector paths and appears as a group in the Layers panel. You can expand the group to work with the individual paths. You can also optimize the group to create a single optimized graphic file from the group of paths.
6. To change the stacking order of objects within the same layer or group, you can drag rows in the Layers panel or use the Arrange commands in the Modify menu.

# Lesson 5: Working with Pages and States

The Flash Catalyst Banner project was created by importing a design from a single-artboard Illustrator file. All of the artwork is stacked together in one page state. In this lesson, students will add new pages and use the Layers panel to show and hide artwork in each page.

## Goals for this lesson

After completing this lesson, students will be able to:

- Explain the difference between page and component states
- Create and name states
- Add and delete states
- Modify states by showing and hiding objects
- Share objects to states
- Remove objects from states

## Page and component states

This lesson begins by defining Flash Catalyst pages and states. Technically a page is a state, but it helps to understand the difference between the main application pages and the states within a component.

### Additional points for discussion or clarification:

Some new Flash Catalyst users react negatively to the 20-state limits for main application pages or states within a component. This is described in the Lesson 5 sidebar, "Less is more." Use this limitation as an opportunity to reinforce the practice of using fewer application pages and more components, including nested components. The result will be more interactive applications that perform better at runtime.

## Duplicating and modifying page states

In this topic, students duplicate existing pages states, assign descriptive names to pages, and modify pages by showing and hiding the artwork in each page (using the Layers panel).

### Key skills or concepts to reinforce:

- By duplicating a page state, you automatically share all of its artwork to the new page.
- You can use the Layers panel to customize the appearance of each page by showing and hiding objects.
- The same objects can have unique properties, such as size, position, and opacity, in each page or state in the same component.
- It's important to give descriptive names to each page or component state.

### Additional points for discussion or clarification:

- A common mistake when naming pages or component states is to enter a space when entering the new name. Flash Catalyst doesn't provide any feedback if you enter an invalid name, instead the state maintains its old name.
- The Lesson 5 sidebar, "Sharing is good" describes how to share the same object to multiple states. Some students will read this and begin playing around with the feature before completing the steps in the current activity. Make sure students read the sidebar for reference and hold off on trying the technique until they complete Lesson 5. They'll have plenty of practice with sharing objects to other states (and making objects the same in all states) throughout this course.

## Adding and deleting states

In this topic, students learn to create a new blank page and add new artwork. They also learn to delete states in the Pages/States panel.

### Key skills or concepts to reinforce:

- One drawback of Adobe Flash Catalyst CS5 (1.0) is that you can't insert a new page (or component state) between or before existing states. Every new state is added after the new state. You cannot drag and drop states to rearrange them. Hopefully these features will be added in the next Flash Catalyst release. For now, it's important that you plan ahead and add or duplicate states in the order they will appear in the final application.
- When you add new objects to the artboard, they appear in a new row in the Layers panel. New objects are added within the Target layer, which is indicated by a light blue highlight in the Layers panel.

### Additional points for discussion or clarification:

- When deleting states, make sure students know not to press the Delete key. Pressing Delete removes selected items in the current state. Flash Catalyst includes four Delete icons (trash cans) that all look identical. Make sure that students are using the correct icon when deleting states in the Pages/States panel.
- In the subtopic, "Adding objects to blank page states," some students might be confused by step 1, which instructs them to show the TopGraphic row. There is a layer folder (the parent layer) with the same name. At first the TopGraphic row is dim, indicating that its contents are not present in the current page (Gardens). By showing the TopGraphic row, the graphic is automatically shared to the current state.
- When importing the new Photoshop artwork (in step 7), you can also have students look in the Library panel to see how the new artwork is stored in its own subfolder in the Images category of the library.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book.*

1. How do you view, add, or delete states in a custom component?
2. How do you insert a new page before an existing page in the application?
3. How do you create two unique pages from one existing page?
4. How do you share an object in an existing page to another page where the object is not present?

5 If you change the properties of an object, how can you quickly apply those properties to the same object in all other states?

6 How can you remove an object from a specific state without removing the same object from other states?

## Answers

1 To view the different states within a component, you can double-click the component in the artboard to view its states in the Pages/States panel and edit the component in Edit-In-Place mode.

2 You can't. Every new state is added after the last state.

3 You can duplicate an existing page and then show, hide, or remove artwork in each page to create two unique pages. You can also add a new blank page, and then share selected artwork from the exiting page to the new page.

4 Select the target page (or components state) in the Pages/States panel. Show the row for the object by turning on its eyeball. Showing the object automatically shares it to the current state. You can also select the object in the source page/states, and then choose States > Share To State and select the target state where you want to add the object.

5 Select the object that has the desired properties and choose States > Make Same In All Other States.

6 You can select the object in any state, and then choose States > Remove From State and select the state from which to remove the object. You can also select a state in the Pages/States panel, select any object, and press the Delete key to remove the selected object from only the current state.

# Lesson 6: Creating Interactive Components

In Lesson 6, students get their first practice creating and modifying interactive components, such as buttons and scrollbars. They begin with ready-to-use wireframe components and then convert custom artwork to one of several types of built-in Flex components. Finally, students add interactivity, such as page navigation, UI controls, and links.

## Goals for this lesson

After completing this lesson, students will be able to:

- Use wireframe components for rapid prototyping
- Convert artwork to built-in components
- Create custom navigation and toggle buttons

- Create custom scrollbars and scrolling panels
- Build custom components
- Add and format text
- Nest components within other components
- Modify components using Edit-In-Place mode
- Share artwork between component states
- Set component properties
- Add and delete Flash Catalyst interactions

## The building blocks of RIAs

This topic introduces Flash Catalyst components and provides an overview on how to create them, edit them, and bring them to life using interactions.

### Key skills or concepts to reinforce:

- The components you create in Flash Catalyst are Flex components.
- You can create components with generic skins by using the wireframe components.
- You can also create components using custom skins by converting artwork to one of several built-in component types.
- You can create custom components with up to 20 states by converting artwork or by nesting other components.

## Using wireframe components

In this topic, students build a very simple wireframe prototype for an application interface. The interface includes wireframe buttons, text input fields, check boxes, radio buttons, and a scroll bar. Refer students to the sidebar on Scroll bars and sliders for a comparison and guidelines for when it's most appropriate to use each.

### Key skills or concepts to reinforce:

- Wireframes can be used as is or modified. If you use a wireframe as is, it is not added as a new asset in the library because the same component is already available in the Wireframe Components panel.
- Wireframes are an excellent tool for quickly generating functional application interface prototypes.

### Additional points for discussion or clarification:

Many students accidentally double-click a wireframe component and find themselves in Edit-In-Place mode. Even if they don't change the wireframe, they have just created a custom skin in the library. The new component appears in the Custom Component category in the Library panel. The wireframe they dragged to the artboard is replaced by the new custom component. Because it's a custom component, it cannot be sized using the Properties panel.

## Converting artwork to built-in components

Students convert artwork into several built-in component types, including navigation buttons, toggle buttons, scroll bars, and a scroll panel.

### Key skills or concepts to reinforce:

- The built-in component types have predefined states and behaviors, such as the Up, Over, Down, and Disabled states of a button. These cannot be changed.
- When you convert artwork to a built-in component type, what you're really doing is adding your own artwork as the *skin* for the component. Flash Catalyst offers designers an easy way to skin Flex components without authoring any code.
- Components cannot be sized using the Height and Width properties in the Properties panel. You must edit the parts of the component to change the overall size of the component. This is also true for custom components.
- Some components require that you define specific parts. For example, a scroll bar (or slider) must have a track and a thumb. A scroll panel must have something (such as a block of text or group of images) defined as the scrolling content. A data list must have a repeated item. If the component has special requirements, a message appears in the HUD with instructions on how to complete the component. The commands to create and edit components are also available from the Modify menu.
- If you create a component from artwork that is shared in multiple states, you lose stateful properties or transition effects associated with that item. This is also true for custom components. It is a good practice to define component structure before building transitions. If you plan to convert artwork to components, avoid sharing that artwork to other states until you have created your components.

### Additional points for discussion or clarification:

- Students may ask about creating Data List components. A data list is a complex type of component that is covered in Lesson 10.
- Moving the parts of a component while in Edit-In-Place can change the size of the component. If you have Auto Size Component Bounds selected in the Modify menu, the component bounds resize automatically in all states. When Auto Size Component Bounds is turned off, you can select Clip To Component

Bounds in the Modify menu. Clip To Component Bounds clips any parts that extend beyond the component.

Students may have questions about the component properties that appear in the Properties panel. Refer to the following when answering questions about setting component properties:

- **Tab Index** controls the order of items when pressing Tab to move the keyboard focus. Lower numbers place the item earlier in the sequence. The -1 indicates a default order based on the Layer panel order.
- **Tab To Focus** enables you to give the component keyboard focus by tabbing to it. If Tab To Focus is deselected, pressing Tab does not give the component keyboard focus. It must be clicked to get focus.
- **Tooltip** displays text as a tool tip when the user interacts with the component.
- **Display As Password** displays the text field's contents as a series of asterisks.
- **Editable** determines whether text in a Text Input can be selected, but cannot be edited (when deselected).
- **Max Characters** automatically sizes a Text Input control to accommodate a specific number of characters. This property is only applicable for a wireframe text input that has not been manually resized.
- **Selected Index** controls which item is selected in a list by default. The first item is 0, the second item is 1, and so on. A value of -1 means nothing is selected.
- **Focus Ring** is the color of the halo or highlight shown on a control when it has the keyboard focus.
- **Hand Cursor** displays the pointing hand when someone rolls over the object with the mouse.
- **Accessible Text** is the text describing the object for screen reader technologies.
- **Radio Button Group** affects which radio buttons can be selected. Only one radio button within a group can be selected. Radio buttons are in the same group if this property is set to the same name. Radio buttons are also in the same group if they are grouped, inside the same component, or at the application level.
- **Page Size** controls how far the thumb moves in a scrollbar when clicking in the track.
- **Step Size** controls how far the thumb moves when clicking the arrows. In a slider, Step Size controls how far the thumb moves when pressing the arrow keys.
- **Snap Interval** forces the thumb in a scrollbar to snap in increments rather than moving smoothly. Page Size and Step Size are always forced to be multiples of the Snap Interval.

## Creating custom components

In this topic, students create a very simple custom component with one state. Custom components can include up to 20 states and can even include other (nested) components. Students will have the opportunity to create and work with multistate custom components and nested components in Lessons 8 and 10.

### Key skills or concepts to reinforce:

When using Edit-In-Place, the Layers panel shows the layers for the main application and any component(s) you are editing. You can drag objects between the main application layers and components or between parent and nested components.

## Working with text and scroll panels

In this topic, students make a Scroll Panel component using the text and scroll bar they've already added.

### Additional points for discussion or clarification:

- When creating the text for the Scroll Panel component, students are instructed to minimize Flash Catalyst. Make sure students minimize and not close Flash Catalyst.
- In step 10 of "Create a scroll panel from artwork," students select the temporary text they created inside the text box and delete it. Sometimes students accidentally delete the entire text box. If this happens, make sure the student creates a new text box that extends below the artboard, as they did in step 4 of "Working with text." The copied text (from the .txt file) needs to be pasted inside this text box. The copied text should be visible within a text box that extends below the artboard. This text box will become the scrolling part of the Scroll Panel component.
- In step 2 of "Create a scroll panel from artwork," the Auto Change Warning message may or may not appear for students. If the warning appears, students can click OK to dismiss the message.
- In step 7 of "Create a scroll panel from artwork," if a student's scroll panel text doesn't scroll, the text box probably isn't long enough to require scrolling (as described above). If this happens, have the student exit Edit-In-Place, choose Modify > Revert Component to Artwork, and then adjust the size of text bounding box and recreate the scroll panel from step 1.

## Adding interactivity

Students use Flash Catalyst interactions to add page navigation to button controls. They also add a Go To URL interaction to open a web page in a new browser window. Video playback interactions are covered in Lesson 8. On Applications Start interactions are described in a sidebar. Students create an On Application Start interaction in Lesson 9. Action sequences are described in a sidebar. Students create action sequences in Lessons 7 and 9.

### Additional points for discussion or clarification:

When entering the URL for a Go To URL interaction, students don't have to add the http: in the address. Flash Catalyst adds this automatically after clicking OK in the Interactions panel.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book.*

1 How do you change the width of a horizontal Scroll Bar component that you've created by converting custom artwork?

2 What types of components do not have predefined states and behaviors?

3 What is the required part of a Scroll Panel component?

4 What should you do if you want the same component, such as a custom button, to appear at the main application level and within another custom component?

5 When might you use an On Application Start Component?

6 What is a Conditional List interaction?

## Answers

1 Components (other than wireframe components) cannot be sized using the Properties panel. To change the width of a Scroll Bar component, you can edit the component in Edit-In-Place mode. Make sure that Auto Size Component Bounds is selected in the Modify menu. Use the Select, Direct Select, or Transform tools to change the size of the horizontal track. Or you can change the position of the left and right scroll arrows.

2 Custom/Generic components can have up to 20 states that you define.

3 When creating a Scroll Panel component, you must identify which object, such as a text box or group of images, to use as the scrolling content.

4 You can add two instances of the component to the application. Place one instance in the main application and the other instance inside the component. If the button needs to appear in different pages or different states in the same component, then use one instance and share the component to the other states.

5 You may want a video or SWF movie to begin playing automatically. Or, you can define a default page state by creating an On Application Start interaction that goes to a specific state when the application opens.

6 You can add conditional list interactions to a data list that trigger an action when someone selects a specific item in the list. The list can include images or text. Conditional list interactions are similar to other interactions. The difference is that the conditional setting, When In Any State, becomes When Any Item Is Selected or When A Specific Item Is Selected. If you choose When A Specific Item Is Selected, you enter the number of the item in the list as the condition for the interaction.

# Lesson 7: Creating Transitions and Action Sequences

In Lesson 7, students will be surprised to learn just how easy it is to create smooth, professional looking transitions using Flash Catalyst. In fact, they can accomplish most of this with a single click. This lesson begins by defining transitions and actions sequences, and then students inspect the default transitions that Flash Catalyst generates automatically. Actually, they run the application and notice the apparent *lack* of transitions between pages and states. When they return to the application, they discover that several transitions were created automatically, but it's up to them to apply smoothing, which extends the duration of the transitions and makes them noticeable. Students will also use an action sequence interaction to add special effects to a Toggle Button component. Finally, students learn about the yellow helper effect bars that Flash Catalyst adds to preserve the fidelity of effects as you make changes to a project with existing transitions.

## Goals for this lesson

After completing this lesson, students will be able to:

- Determine when to use transitions and action sequences
- Add smooth transitions between states
- Fine-tune transition timing and other properties
- Add and remove effects in a transition
- Add and modify action sequences
- Recognize helper effects (yellow bars) and preserve transition fidelity

## Transitions and action sequences

To help students get off to a good start with transitions and action sequences make sure they understand the differences between transitions, action sequences, and actions (effects).

### Key skills or concepts to reinforce:

- A **transition** is an animation that plays as the user moves from one page or component state to another.
- An **action sequence** is a type of interaction that triggers one or more actions or effects. Action sequences occur within a single page or component state.
- An **action** is a special effect that occurs during the transition or action sequence. For example, an object moves, fades, or rotates. An object's property changes, such as going from transparent (0% opacity to 100% opacity). A video begins to play or stops.

### Additional points for discussion or clarification:

The State Transitions section of the Timelines panel includes a transition placeholder between every page in the application, even if there is no difference between the two pages (such as two blank pages). As soon as there is a difference between two pages, Flash Catalyst adds an Effect bar to the transition. This Effect bar begins with a duration of zero seconds. For example, if an object appears in page 1 but not in page 2, there is a Fade Out transition Effect bar added between page 1 and page 2. You can't see this effect when your run the project because it happens instantly. To make the transitions noticeable, you need to add smoothing.

## Adding smooth transitions

In this topic, students turn the default zero-second transition effects into smooth looking transitions that are visible when running the project.

### Key skills or concepts to reinforce:

- By extending the duration of the default effect, it becomes apparent. The easiest way to extend the duration of a transition Effect bar is to click Smooth Transition in the Timelines panel.
- To can change the default Smooth Transition settings, open the Smooth Transition Options dialog box.

## Fine-tuning transitions

Students learn to adjust transition timing (when a transitions starts and end) and duration (how long it takes to complete) by using the Effect bars and the Properties panel. Students also learn about *easing* transitions.

### Additional points for discussion or clarification:

The effects of easing properties can be difficult to understand and the subtle differences between settings even harder to recognize. Encourage students to experiment with the different settings and use the Play button in the Timelines panel to examine the differences between properties.

## Adding and changing effects

In this topic, students see how changing the contents in a page can automatically change the type of effects that appear automatically in a transition. For example, if an object appears on one page but not the other, a fade effect is added. If you share the object to both pages, the fade effect is removed. If you then move the object in one page, a move effect is added.

The other method for adding effects is to select a transition, and then manually add additional effects from the Add Action pop-up menu.

## Adding action sequences

In this topic, students add a special type of interaction called an action sequence. The *action sequence* plays one or more effects when the user interacts with an object in the application (such as a clicking button) or when the application starts.

### Key skills or concepts to reinforce:

- Action sequences are interactions triggered by an event.
- Action sequences play within a single page or component state, so they can repeat.

### Additional points for discussion or clarification:

Open a discussion with the class about different types of special effects you can create by using action sequences. Some examples to begin the discussion include:

- Change the properties of an object, such as revealing a hidden image when someone rolls over it (by changing its Opacity property).
- Create buttons that slide, bounce, jiggle, or spin as the user interacts with them.
- Combine multiple actions or effects in the same action sequence, such as fading in a panel as it slides into view and then plays a video or sound effect.

## Flash Catalyst helper effects

In some cases, the changes you make to object that are part of an existing transition can affect the appearance of the transition. For example, grouping objects after they've been used in a move transition can break the transition. To preserve the fidelity of your transitions, Flash Catalyst adds *helper effects*. In this topic, students learn to identify and preserve Flash Catalyst helper effects.

### Key skills or concepts to reinforce:

- Helper effects appear in the Timeline as yellow bars.
- If you change the Delay value of a transition, you must also change the Delay value for the yellow Effect bar to match. The original Effect bar and the yellow Effect bar must stay in sync.

- Do not manually change the duration of a yellow Effect bar. Flash Catalyst will adjust the yellow Effect bar duration if needed.

## Additional exercise (optional)

In the sample Banner project, students work with the default Fade In and Fade Out transitions between pages. This causes the panels at the bottom of each page to fade in and out between pages. Explain to students how they can turn these into Move transitions that slide in and out of view. The steps are:

1. Make *every* panel visible in *all* page states.
2. Position all of the panels below the artboard in every state. To do this, select the Panels group in the Layer panel and press the Down Arrow key several times to slide the panels below the artboard. Then, with all panels still selected, choose States > Make Same In All Other States.
3. Go to each page and reposition its panel (use the Layers panel to select it) where it belongs, at the bottom of that page. For example, in the Design page the Design panel is at the bottom of the banner, but all other panels are still out of view below the artboard.
4. Select every transition by Shift-clicking in the State Transitions section of the Timelines panel.
5. Click Smooth Transitions and then run the project. Flash Catalyst adds smooth transitions that slide each panel in and out of view as you move from page to page.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1. How do you quickly add smooth transitions to every transition in the application?
2. How can you change a Fade transition (fading an image) to a Move transition between two pages?
3. What happens when you select a transition's Effect bar in the Timeline and click the Delete icon (trash can) in the Timelines panel or press the Delete key?
4. What does the Smart Smoothing smooth transition option do?
5. How do you add smooth transitions between two states in a custom component?
6. What do the yellow Effect bars represent when they appear in a transition in the Timeline?

## Answers

1 In the Timelines panel, click the first transition in the State Transitions list. Hold down the Shift key and click the last transition in the list. With all transitions selected, click Smooth Transition.

2 First, you need to share the image to both pages. You can do this by turning on visibility for the image in both pages using the Layers panel. The Fade effect is removed. Place the image in different locations in each page (the same image has different X/Y values in the Properties panel). The Move effect is added.

3 The Effect bar duration changes to 0, but the Effect bar remains in the Timeline.

4 Smart Smoothing adjusts the duration and delay (start time) of each effect, creating a series of staggered effects. The effects play at different times over a duration you specify. Flash Catalyst uses a logical order for effects, beginning with objects fading out. After objects fade out, all resize and move effects play, followed by objects fading in.

5 Edit the component in Edit-In-Place mode. When the component is in Edit-In-Place mode, its transitions and action sequences appear in the Timelines panel.

6 In some cases, changes to objects that are currently part of a transition can negatively affect the appearance of the transition. For example, yellow bars appear when you move a group (or its children) on one side of the transition. Flash Catalyst adds additional effects to adjust for the change and correct the transition automatically. The added effects appear as yellow bars.

# Lesson 8: Adding and Controlling Video and Sound

Before video is added to a Flash Catalyst project, it must be compressed and saved in a file format that's compatible with Flash Catalyst and the web. This process involves encoding the video and saving it in either the FLV or F4V file formats. Lesson 8 begins by providing background information and tips on shooting and preparing video for the web. Students then import two sample videos that have already been prepared for use in the Banner project. Students place one of the videos in the application and then create a custom Play/Pause button using interactions and a Toggle button they created in Lesson 6. Finally, students add sound effects to the top navigation buttons used to move between pages.

## Goals for this lesson

After completing this lesson, students will be able to:

- Prepare video for the web

- Import video and sound
- Preview video and sound
- Add video to an application
- Control video playback
- Add sound effects

## Preparing video for the web

Flash Catalyst supports FLV and F4V video files. It's likely that students will want to know how to get their own videos from a camcorder into Flash Catalyst. This topic provides a very high level overview of the process, including some background information on encoding, video codecs, and tips for shooting web friendly video.

### Key skills or concepts to reinforce:

- Video editing programs, such as Adobe Premiere Pro CS5, are excellent tools for capturing video from a camcorder or videotape deck.
- It's always good to edit your video before adding it to the web. Keep your video clips as short as possible.
- Video must be encoded and saved as an FLV or F4V file before you can import it into Flash Catalyst. Review "What is encoding?" and "What is a video codec?"
- Use presets when encoding video. Most applications, including Adobe Media Encoder, include presets for encoding web video.
- Whenever possible, encode a file from its uncompressed form.

### Additional points for discussion or clarification:

- Make sure that students review the video shooting tips in the sidebar "Shooting web-friendly video."
- Discuss some of the various applications that students can use to encode video, including Adobe Media Encoder, which is installed with Flash Catalyst by default.

For more information on web video, visit www.adobe.com/devnet/video.

## Video encoding demonstration (optional)

If you can set aside an extra five or ten minutes, students will benefit from a short demonstration of encoding video using Adobe Media Encoder. To complete the demo, you need one short video clip. For this demonstration, any short QuickTime or AVI file will work. The steps are:

1 Start Adobe Media Encoder CS5.

2. Choose File > Add (or click Add), browse to locate your source video file, select it, and click Open. This adds a link to the file in the Queue.
3. Click the Format down arrow to open the pop-up menu, and select FLV|F4V.
4. Click the Preset down arrow to open the pop-up menu, and select a web compatible preset, such as FLV - Web Medium (Flash 8 and Higher).
5. Click the Output File link. Enter a name and location for saving the encoded file, and click Save.
6. Click Start Queue. The video is encoded and a progress bar displays at the bottom of the dialog box. When the encoded file is ready, a green check mark appears beside the file in the queue.

## Importing and previewing video and sound

In this topic, students import two video (.flv) files and one sound (.mp3) file. They import all three files together, so the files are added directly to the project library without placing any media in the application.

### Key skills or concepts to reinforce:

- When you import more than one media file, the files are added to the project library, but not placed in the artboard of the application.
- If you import a single video file, a new video player object is added to the application automatically.
- All sound files are added directly to the library, even when you import a single file. This is because sound files are not added directly to the artboard. Instead they are added as an effect in the Timeline.

## Adding video to the application

Students add the restaurants.flv video file to a video player inside the Image Slider component. This video is added to the Restaurants page state.

### Additional points for discussion or clarification:

- When you add a video file to the application (artboard), you are creating a video player object. Just as with images and other objects, the same video player can be shared to other states. If you want the video to appear in different pages or states in the same component, share it to those states instead of adding new instances.
- As with other images and objects, the same video player can have different properties in each state. You can change these properties in the Properties panel. Some properties, however, such as Auto Play, Loop, and Mute, persist across all states for the same video player.

- Even though students add the video inside a component, they could just as easily place the video directly in the main application layers.
- In the "Set video player properties" section of this topic, there are explanations of the various Scale Mode properties for a video player. If time allows, you can have students try these or you can demonstrate them. If students try these on their own, make sure they click Edit > Undo to return their video to its default dimensions.

## Controlling video playback

In this topic, students replace the default video controls with the Toggle Button component they created in Lesson 6. To do this, they need to move the toggle button inside the Image Slider component. Then, they add the video player and the toggle button to a new custom component with two states, VideoPaused and VideoPlaying. This allows them to change the appearance of the toggle button while the video is playing. They make it semi-transparent. With the video player and toggle button together in the same component, they add interactions to the toggle button that control the playback of the video.

### Key skills or concepts to reinforce:

- A component or group can be assigned video control interactions, but must be in the same location (in the main application layers or inside the same component) as the video player you want to control.
- You can use the Layers panel to quickly drag objects between components and the main application layers, or between a parent component and its nested components.
- If you want to use the same control in two locations, such as at the main application level and inside a component, you can use two separate instances of the same control.

### Additional points for discussion or clarification:

- Students place the video inside the Image Slider component. Some students may be confused when they edit the component because the actual name of the Image Slider is CustomComponent1 (in the library). Image Slider is the name they gave the component in the Layers panel. Explain to students that they can also take the extra time to rename the component in the library so that it matches what it's called in the Layer panel. You should also explain, however, that component names in the library cannot contain spaces.
- You might think that it would be easier to simply turn on Auto Play for the video, but doing this would cause the video to begin playing when the application starts, not when you navigate to the page that includes the video. In this project the Image Slider component appears on every page. This is the

parent component for the nested components holding the videos. So even though you don't see the videos in the first page (Feature), it's present (but positioned off the artboard). With Auto Play selected, the video would play when the application starts. You would not see the video in the Feature page, but you would hear the audio.

## Adding sound effects

In this topic, students add the Flash Beep.mp3 and button_click.mp3 files as sound effects to the top navigation buttons. If time allows, you can have students practice what they've learned by adding the same sound effects to the Back and Next navigation buttons. Remember, there are two versions of the Back button and two versions of the Next button.

### Key skills or concepts to reinforce:

- Sound effects are added in the Timelines panel.
- You can add sound effects to objects in a transition or action sequence

### Additional points for discussion or clarification:

- The location of the sound effects library/directory is different in Windows and the Mac OS. When adding the sound effect, Flash Catalyst should default to this directory. If students are having trouble locating the Flash Beep.mp3 file, they can use the copy that is provided in the *Adobe Flash Catalyst CS5 Classroom in a Book* Lesson08 folder.
- Students may ask why they are adding the sound effects to the buttons in an action sequence and not adding the sound effects to transitions. If you were to add the sound effects to the objects in a transition, the sound would play during the transition only. This means the roll over effects would not work. You would also need to add the sound effect to every transition, as opposed to adding the sound one time to the button.
- The default duration of a sound effect in the Timeline is one second. This is fine for most short sound effects, such as button clicks. If the sound effect is longer, such as a short dialog clip, you will need to change the duration of the Effect bar.
- For longer audio files, such as narration, you can use a sound-only FLV file. When you do this, you may want to hide the video player controls or place the video player outside the artboard.

## Additional Exercise (optional)

The Banner project also requires a second video, cityscape.flv. The second video appears in the Future page state. The cityscape.flv file replaces the Future Image (still image) in the Image Slider component (CustomComponent1). Students imported the cityscape.flv in this lesson, but they do not complete the steps to add the video to the Image Slider component.

If time allows, you can have student practice what they've learned by adding the second video. They cannot reuse the same video player to hold the cityscape video, however, because both videos need to be visible at the same time in the Image Slider component.

The following is a summary of the steps for adding the, cityscape.flv video file.

1 Select the Future page (the last page) in the Pages/States panel, and edit the Image Slider component in Edit-In-Place mode.

2 Replace the Future Image with the cityscape.flv file, and hide the default controls.

3 To control the playback of the cityscape.flv video, students will need to add a new instance of the Toggle Button component (from the library). Create a new custom component that contains the video player for the cityscape.flv video and the new toggle button. Name the new component Future Video, and create the VideoPaused and VideoPlaying states.

   If desired, students can modify the appearance of the toggle button while it is playing (in the VideoPlaying state).

4 Add the interactions to play and pause the video. Be sure to set the Toggle Button component as Selected when in the VideoPlaying state to reset the toggle button.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1 How do you use the same video player to play a different video file in another state?

2 If you can use the same video player to play different videos, then why can you not use the same video player to play both the restaurants.flv and cityscape.flv files when viewing the Restaurants and Future pages?

3 What does encoding refer to?

4 Why can't you use the Auto Play property to play the videos in this project without adding controls?

5 How can you quickly move an asset from the main application into a component without adding a new instance of the asset?

6 In this lesson, why did you add the sound effects to an action sequence instead of adding them to the buttons within a transition?

## Answers

1 If a video player is in more than one page state in the main application or in more than one state in the same custom component, you can use the Source property in the Properties panel to control which video plays in each of those states.

2 First, the restaurant.flv and its control, the Toggle Button component, are both inside a new custom component named Restaurants Video, which is nested inside the Image Slider. This is done so that you can create separate VideoPaused and VideoPlaying states. When you add the cityscape.flv file to the Future page, it will also go inside a new custom component within the Image Slider. Each video is in its own custom component and cannot use the same video player. Also, the videos have different positions within the Image Slider component, and both videos need to be visible at the same time as the Image Slider changes position.

3 In Flash Catalyst, encoding involves converting your final edited video footage to a format compatible with Adobe Flash Player. Flash Catalyst supports FLV and F4V Flash video files. FLV and F4V are container formats for Flash Video.

4 Selecting the Auto Play property for the video player will cause the video to begin playing automatically when the application starts. This application begins on the Feature page and the video appears on that page (in the Image Slider), even though the video is positioned out of view in that state.

5 Edit the component in Edit-In-Place mode. In the Layers panel, drag the asset from the main application layers into the layers for the component that you're editing.

6 If you were to add the sound effects to the objects in a transition, the sound would only play during the transition. This means the roll over effects would not work. You would also need to add the sound effect to every transition, as opposed to adding the sound one time to the button.

# Lesson 9: Integrating SWF Content from Other Creative Suite Tools

A lot of really good (and not so good) SWF content is available for use in projects—from such proprietary files as animated logos, banner ads, and marketing presentations to large collections of royalty-free content. In Lesson 9, students learn that not all SWF content is the same and not all is equally compatible with Flash Catalyst. Students will import a couple of SWF file examples, including an animated logo and a multiscreen site. In reality, you'd probably build the second example (traveler.swf) entirely in Flash Catalyst, but it works well as an example of existing content that you can add to your application and control using interactions. Students use an On Application Start interaction to control the movie. First, they play the movie from frame 1, and then use a Go To Frame And Play effect to play the SWF file from a specific frame. Finally, students learn to create an animated button or custom components by incorporating existing SWF content.

## Goals for this lesson

After completing this lesson, students will be able to:

- Use SWF files in Flash Catalyst projects
- Add SWF files to an application
- Control the playback of a SWF file
- Play a SWF file from a specific frame
- Create an animated button using a SWF file

## SWF files

Students who are familiar with Adobe Flash and SWF files may be tempted to skip this first topic, but assure them it contains some important information about SWF files as they relate to Flash Catalyst.

### Key skills or concepts to reinforce:

- Several applications can produce SWF files, but only ActionScript 3 SWF files published using Adobe Flash Professional can be controlled in Flash Catalyst.
- If the SWF file you want to add to Flash Catalyst includes Flash TLF text, you need to merge the shared library into the underlying code.

### Additional points for discussion or clarification:

- If your SWF file loads video (or other content) dynamically, you need to copy the video (or other content) to your published Flash Catalyst project folders. You can do this only after you publish the project, which means you first import the SWF file into your Flash Catalyst project, then publish the project. Publishing the project creates the project's main.swf file. You can then copy the video into your published folders. The video needs to maintain the same position relative to the Flash Catalyst main.swf file that it had relative to the SWF file you're importing. It may help to illustrate this for students by using a white board.
- *Adobe Flash Catalyst CS5 Classroom in a Book* includes instructions for merging a shared library in Flash Professional CS5, but not CS4. You can use Flash Professional CS5 to open the SWF file created in Flash Professional CS4 and use these steps to merge the shared library. Another option is to convert the TLF text to Classic Text.

## Adding SWF files to Flash Catalyst

Students begin this topic by importing an animated logo (SWF file) to the artboard.

### Key skills or concepts to reinforce:

You cannot preview a SWF file in the Library panel. You need to run or publish the project to test the SWF file.

### Additional points for discussion or clarification:

When students learn that you cannot preview a SWF file in Flash Catalyst (in the Library panel), they may be confused because they just imported an animated logo and could see the animation playing in the artboard. They can see the animated logo because the SWF includes a movie clip that plays (repeatedly) in the first frame of the SWF movie's main timeline. In most cases you will see only the first static frame of the SWF movie and must run the project to see the animation.

## Controlling the playback of SWF files

In this topic, students import the traveler.swf file and add it to a Flash Catalyst project. The purpose of this topic is to show how Flash Catalyst effects can be used to control the playback of a SWF file. You can add the action sequence used to control the SWF file to a component or grouped artwork. In this topic, students add SWF control to an On Application Start interaction, so the SWF file begins playing as soon as the application starts.

### Key skills or concepts to reinforce:

To control the playback of a SWF file, add an action sequence and then choose Add Action > SWF Control, and select the type of control effect you want to add (such as Play, Stop, or Go To Frame And Play).

### Additional points for discussion or clarification:

With this first release of Flash Catalyst CS5, there are still some issues when running projects with SWF files in certain browsers and Flash Player versions. If, when running the Lesson09_Traveler.fxp project, the SWF does not begin to play properly, try clicking the Refresh button in your web browser. Or, close the browser, return to Flash Catalyst, and run the project again. This usually fixes the problem.

## Adding a SWF file to a component

In this topic, students learn how to add animated SWF content to a component by adding a simple animation that plays during the Over state of a button. This is done in three main steps.

1. Add SWF playback controls between the Up and Over states.
2. Add a Play SWF control to play the SWF animation when someone rolls over the button.

3 Add a Stop SWF control to stop the animation from playing when the mouse moves away from the button (returning to the Up state).

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1 When can a SWF file be previewed without running or publishing the project?

2 What happens when you import a collection of SWF files into Flash Catalyst?

3 What is the preferred application for creating SWF content that you plan to add to a Flash Catalyst project?

4 Can a Flash Catalyst project include a SWF file that loads content, such as a linked video file, dynamically at runtime?

5 Can you round-trip edit a SWF file that you've added to a Flash Catalyst project?

6 How do you add an On Application Start interaction used to control a SWF file?

## Answers

1 If the entire movie plays in the first frame of the SWF file's main timeline, you will see it when the SWF file is added to the Flash Catalyst artboard.

2 When you import more than one SWF file at the same time, the SWF files are added to the Images category in the Library panel. No content is added to the artboard.

3 Adobe Flash Professional CS5 is the preferred application for creating SWF files for use in Flash Catalyst.

4 Yes, but the linked content needs to maintain the same position relative to the Flash Catalyst main.swf file that it had relative to the SWF file you're importing (the SWF file that loads the linked content).

5 No. There is no integration between Flash Catalyst and Flash Professional. If you need to edit the SWF file, make your changes in Flash Professional, republish, and import the new file into Flash Catalyst. Then, select the SWF Asset in your application and use the Source link in the Properties panel to swap the old SWF file for the new one you've added to the library.

6 If you want the SWF file to begin playing automatically when the application starts, then make sure nothing is selected in the artboard. When you click Add Interaction, the interaction will default to On Application Start. Next, add an action sequence interaction. In the Timelines panel, select the new On Application Start action sequence, select the SWF asset in the artboard, and use the Add Action pop-up menu to add a SWF Control, such as Play SWF.

# Lesson 10: Designing with Data

The Data List component is a special type of Flex component that lets you build traditional (spreadsheet style) and nontraditional looking lists of data in Flash Catalyst. You can build a fully functional and ready to use data list using content (images and text) that you store in the project file. You can also build a functional Data List component that includes a sample collection of records (design-time data) used to demonstrate how the list should look and function. A developer can then leverage the Data List component you create in Flash Catalyst (without needing to recreate it in Flash Builder) and replace the sample data by binding the component to an external data source.

In this lesson, students are introduced to the Flash Catalyst Data List component and design-time data. They create a Data List component using sample design-time data. Finally, they see how easy it is to build a traditional looking data list by using the wireframe Data List component.

## Goals for this lesson

After completing this lesson, students will be able to:

- Design visually interesting interface components for browsing and displaying large collections of data
- Create a Data List component using images and text
- Set data list properties
- Use design-time data to demonstrate the appearance and behavior of a data list
- Add a wireframe Data List component

## Flash Catalyst data lists

Flash Catalyst Data List components and design-time data are defined in this topic. Before students build a new data list, they open an existing project and view an example of the data list they will create.

### Key skills or concepts to reinforce:

- You can build a completed Data List component in Flash Catalyst that doesn't require additional development in Flash Builder. Design-time data can be your "real" data, but the final content (images and text) needs to be stored within your project file. If this content changes, you must republish and redistribute the Flash Catalyst project.
- Most data lists are used to present a large number of records that change over time. You create the visual appearance and design the behavior of the list in Flash Builder. A Flex developer uses Flash Builder to bind the Data List component to a real data source.

- A Flash Catalyst Data List component doesn't need to look like a spreadsheet. It can include both text and images. It can look like a scrolling list of records, or a tiled wall of images and text.
- In a Flash Catalyst Data List component, each record has the same Normal (up) and Over states. You define the appearance of these in Flash Catalyst by designing a repeated item template (a required part of every Data List component).

### Additional points for discussion or clarification:

When students preview the data lists in the Restaurants_finished.fxp document, draw attention to the fact that the project includes only three main application pages. As the user navigates this application, there are several clearly unique screens and even more views as the user browses the list of restaurants, reads restaurant reviews, or changes between Reviews and Review Stats. This depth of content is creating by using custom components with multiple states, by nesting components inside other components, and by using data lists to present a large number of records. The sample restaurants in the list and the sample reviews are design-time data. These records can be replaced by a developer when they bind the lists to a real data source.

## Creating a Data List component

In this topic, students recreate the restaurant reviews data list. To do this, they place the new data list inside the Review_Popup custom component, which is nested inside the SupPages custom component, which appears in the Sup Pages page state.

### Key skills or concepts to reinforce:

The main steps to creating a Data List component are:

- Design a sample record to use as the repeated item, and add a scroll bar (if needed—not all lists require scrolling).
- Select the sample record, the scroll bar, and any other background artwork you want to include in the data list. Convert the parts into a new Data List component.
- Select the collection of parts (text, image, other artwork) that represent a single record in the list. Define these elements as the "repeated item" for the list.

### Additional points for discussion or clarification:

- Before creating the data list, make sure that students understand the concepts of a design-time data and the repeated item template. The repeated item template is explained in more detail in the subtopic "Define the repeated item." Still, you may find it helpful to explain what this is before they start to build their first component.

- Possibly one of the biggest challenges for students in this task is making sure they add the new data list in the correct location. Tell students to complete steps 1 through 6, and then make sure everyone has the correct component (Reviews_Popup) in Edit-In-Place mode before continuing.
- When Converting the sample record into a Data List component, make sure that students select all of the artwork for the sample record *and* the scroll bar. The title “Convert the sample record into a Data List component” can be misleading because the scroll bar is not part of the sample record. It is part of the Data List component and should be included in the selection in steps 1 and 2.
- When defining the repeated item, students should *not* select the scroll bar (as noted in step 2).

## Setting data list properties

In this topic, students begin by adjusting the bounding box for the data list. The bounding box defines the visible area of the list. This is usually the same size as the scroll bar (if the list needs to scroll), but it doesn’t have to be. It should however, be kept within the visible area of the artboard. Students also explore layout options for the repeated item template. Make sure students return to the Vertical layout option before continuing. Finally, students edit the appearance of the repeated item template by modifying the Over state.

### Key skills or concepts to reinforce:

Editing the repeated item edits the appearance and behavior of every item in the list. Even after you bind the component to real data, your records will appear as you design them using the repeated item.

### Additional points for discussion or clarification:

- You cannot add, remove, or duplicate states in the repeated item template, but you can add, remove, and modify the artwork in each of the default states (Normal, Over, and Selected).
- A Data List component can have the same properties as other components and these properties can change from one state to the next. For example you can change its opacity and position, so that when you move from one page to the next, the data list can slide over to make room for new content in the page. You can add other properties, such as adding a drop shadow filter.

## Using design-time data

Design-time data is the sample content used to represent the look and behavior of a Data List component. When you first define the repeated item for a data list, Flash Catalyst adds five new records to the Design-Time Data panel. Each record is an exact copy of the repeated item. In this topic, students replace the images and text in the five sample records with a new content to give the list a more realistic appearance. Students will also add additional sample records to demonstrate the functionality of the scroll bar.

### Key skills or concepts to reinforce:

- If your design-time data is just placeholder data and you plan to bind the content to a "real" data using Flash Builder, then you can reuse some of the same images and text to create multiple records.
- If you are creating a finished data list, all images in the list must be stored in the project library. To update records, you will need to edit them using Flash Catalyst and republish and redistribute the project.

### Additional points for discussion or clarification:

If you look in the Design-Time Data panel, you'll notice that records begin with row zero, not the number 1. This creates the potential for errors when replacing design-time data with sample records that you've named using a numbering system that begins with the number 1. For example, you may have created several design-time images named photo_01, photo_02, and so on. If you know that you're creating images for design-time data, it helps to name your sample content in sequential order beginning with zero.

## Use a wireframe data list

In this topic, students add a wireframe Data List component to the artboard in a new blank Flash Catalyst project. This is simply to demonstrate the look and behavior of the wireframe Data List component. Explain to students that they can use the wireframe Data List component as is or edit it.

When the students edit the component in Edit-In-Place (in step 3), have them look in the Library panel and notice they've just added a new custom component to the library. Editing a wireframe creates a custom skin.

## Additional exercise (optional)

By default, a data list opens with nothing selected. You can set the list to open with a specific record selected. If time allows, you can demonstrate how to do this or have students practice this on their own. The steps are:

1 Open the Restaurants_finished.fxp file in the Lesson10 folder.

2 Select the Home page in the Pages/States panel.

3 In the Layers panel, unlock the Home Content layer, select the Home Content custom component, and click State1 in the HUD to enter Edit-In-Place mode.

4 In the Layers panel, select Data List.

5 Expand the Components section in Properties panel. The Selected Index property is set to -1 by default. This means that no record is selected when the application runs. Change the Selected Index property to 2 (for row number 2 in the Design-Time Data panel). Notice that the third restaurant, in the list shows its Selected state, as defined in the repeated item template. You can modify the Selected state's appearance by editing the repeated item.

6 Run the project. The project opens with the third restaurant selected.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book.*

1 What is a drawback of creating the entire finished data list in Flash Catalyst, without binding the component to an external data source?

2 What are the three main steps to creating a Data List component?

3 What is the difference between data list repeated item Spacing and Padding in the Layout section of the Properties panel?

4 How can you visually tell users which item is currently selected in the data list?

5 How can you set a data list to open with a specific record selected?

6 What is created automatically when you edit a wireframe Data List component?

## Answers

1 Any images used in records must be stored in the project library. To update records, you will need to edit them using Flash Catalyst and then republish and redistribute the project.

2 First, create and position the artwork and/or text for the repeated item (one record). Then select the objects in the database, including the artwork for the repeated item and the scroll bar (if one is required) and convert the selected items into a Data List component. Finally, edit the Data List component, select the parts for the repeated item (not the scroll bar) and convert them to the repeated item data list part.

3 Spacing is the distance between each item in the list. Padding refers to the space between the list items and the bounding box.

4 Edit the Selected state in the repeated item template. Add a border, graphic, transparent overlay, or some other treatment that visually communicates that a record is selected.

5 Select the Data List component and in the Properties panel, change the Selected Index property to the row you want selected when the application runs. Keep in mind that the first record in the database is row zero.

6 When you edit any wireframe component, including a wireframe Data List component, Flash Catalyst adds a new component in the Components category of the Library panel. Changes you make to the component are defined in the new custom component.

# Lesson 11: Drawing and Editing Artwork

Don't let the title fool you. Flash Catalyst doesn't claim to be a full-fledged drawing program, but it does provide a good set of basic tools for adding basic shapes and lines. You'll find what you need for simple editing, adding basic interface elements, and rapid prototyping. For more advanced editing, you can launch and edit your Flash Catalyst artwork in Adobe Illustrator or Adobe Photoshop.

In this lesson, students begin by exploring the Flash Catalyst Tools panel. After that, there's an overview of using the rulers, guides, and grid. Students use the drawing tools to create a simple user interface panel. They enhance the appearance of their drawings by adding filters. Finally, students learn how to launch and edit artwork using Illustrator and Photoshop.

## Goals for this lesson

After completing this lesson, students will be able to:

- Use the rulers, guides, and grid
- Draw basic shapes and lines
- Change stroke and fill settings
- Add and modify gradient fills
- Group and position objects
- Transform shapes and lines
- Apply and remove filters
- Launch and edit artwork in Adobe Illustrator CS5
- Launch and edit artwork in Adobe Photoshop CS5

## Flash Catalyst drawing tools

In this topic, students learn which tools are available in the Flash Catalyst tools panel.

### Key skills or concepts to reinforce:

Some tools share the same space in the Tools panel. To view a hidden tool, hold down the mouse button to open the pop-up menu and select a new tool.

### Additional points for discussion or clarification:

There are two selection tools in Flash Catalyst. Both tools usually work fine for selecting items in the artboard, but there are times when you need to make sure you're using the correct tool for the job. Anytime you need to select or manipulate an individual graphic or path that's part of a group, make sure the Direct Select (light arrow) tool is active. Otherwise you will select the entire group, not its individual parts. Using the Selection tool (dark arrow) to click grouped artwork in the artboard has the same result as clicking the group (parent) in the Layers panel. It selects the entire group, including all of its children. Using the Direct Select tool is the same as clicking individual rows (group children) in the Layers panel.

## Using rulers, guides, and the grid

In this topic, students show and hide the rulers, guide, and grid. They also set grid properties and add guide lines to define the size of the panel they're going to draw.

Most designers are familiar with the use of rulers, guides, and the grid, but it's important that all students complete this topic; don't let anyone skip it. This is where they set the guides and grid used for drawing throughout this lesson.

### Key skills or concepts to reinforce:

- Rulers are used for precise measurement of objects as you draw them.
- The grid is used for precise drawing and alignment of objects.
- Guides are custom horizontal and vertical lines that you place in the artboard to use as guides as you draw.
- Guides and gridlines do not appear in the published application.

### Additional points for discussion or clarification:

In this topic, students set guides in the main application. These guides do not appear when editing a component in Edit-In-Place mode. You can add separate guides that are unique to each component. This includes components that are nested inside other components. Each component has its own guides. The grid, on the other hand, is the same in the main application and all components. It's either on or off.

## Drawing basic shapes and lines

Students will create a very simple user interface element using basic shapes and lines. Completing this topic doesn't require any artistic talent. Each subtopic and its steps are designed to teach and reinforce the fundamental skills for using the drawing tools. The shapes are drawn separately and then assembled and enhanced to create the panel.

**Additional points for discussion or clarification:**

- The panel header has rounded top corners and square bottom corners. This is created by creating a rounded rectangle (Panel Header) and then placing a smaller rectangle (Header Overlay) over the bottom third to hide the rounded corners. Students draw the Panel Header above the panel and the Header Overlay below the panel just so they can see the parts as they draw them. They will position the parts later in this lesson.
- In the subtopic "Draw basic lines," students draw two crossed lines inside an ellipse. These shapes will be grouped and rotated to create the Close button. It's important to draw both lines (horizontal and vertical) in the exact center of the ellipse and make them intersect at their exact center points. Students can use the grid lines to do this, as shown in the illustrations. If the lines are not perfectly symmetrical, they may not scale properly when students scale the button during "Grouping and transforming."

## Changing stroke and fill

In this topic, students stylize the parts of the panel by adding stroke and fill. The panel header receives a gradient fill.

**Additional points for discussion or clarification:**

- When changing fill rotation in the Properties panel, you need to have the Fill properties expanded. You can click the small arrow beside the Fill property to expand it. This reveals the Fill Rotation settings. You can change the rotation value by entering a new value, dragging the pointer left or right (scrubbing) across the value field, or by dragging in the Fill Rotation icon. If you center-click on the Fill Rotation icon, the gradient changes by 180 degrees. For example if the Fill Rotation value was set to 0, then center-clicking the Fill Rotation icon changes the gradient to 180 degrees. It basically flip flops the gradient to its exact mirror image.
- The Panel Header and Header Overlay shapes both have the blue gradient fill applied. These shapes are different sizes so the gradient rotation doesn't line up perfectly when you overlap the shapes. Students correct this by adjusting the gradient rotation value in the Header Overlay. If, after adjusting rotation, the gradients still don't line up, students should check the size of their shapes. The shapes may be a little out of spec. Changing stroke also affects shape size. The size of the Panel Header shape should be 398 x 59 (width by height). The Header Overlay shape is 398 x 19.

## Grouping and transforming

In this topic, student group the Panel Header and the Header Overlay so they can be moved as a single object. Students also group the ellipse and the intersecting lines used for the Close button. After grouping, students use the Transform tool to rotate the Close button (turning the plus into an X). Finally, they size and position the Panel Header and Close button.

### Key skills or concepts to reinforce:

- Groups have their own set of properties in the Properties panel, which makes it easier to size, position, and transform drawings with lots of smaller parts.
- You can use the Select tool to select, transform, size, and position grouped artwork. Use the Transform tool to rotate a group.
- You can use the Direct Select tool to select and manipulate the parts of a group after grouping.

### Additional points for discussion or clarification:

In step 5 of "Transform shapes" make sure the Rotation value is exactly 45 degrees (or -45). This creates a perfectly symmetrical X inside the ellipse. If the X is not symmetrical, then sizing the shape will exaggerate the lack of symmetry and will not result in a very appealing Close button.

## Applying and removing filters

Flash Catalyst includes a collection of filters that you can add to images, shapes, components, videos, and other assets. The available filters include Blur, Drop Shadow, Inner Shadow, Bevel, Glow, and Inner Glow. Each filter has a set of properties for customizing the appearance of the filter.

In this topic, students add drop shadows and bevels to the parts of the panel they drew.

### Key skills or concepts to reinforce:

- Filters that were applied in the design document, flattened, and then imported to Flash Catalyst are not editable in the Properties panel.
- Supported filters that were applied in Illustrator are editable in Flash Catalyst, as long as you choose the Filters > Keep Editable fidelity option during import. The supported filters you applied in Illustrator, such as drop shadows, will appear in the Properties panel as if you added them in Flash Catalyst.

### Additional points for discussion or clarification:

- When students add and modify the Bevel and Drop Shadow filters, they are instructed to set values for Distance, Blur, Angle, and Strength. Many of these options may be set correctly already. It's okay if students want to experiment with the filter settings. Doing so will not throw off any steps moving forward, but it will change the appearance of their drawings.

- You can use overlapping shapes to create the illusion of a cutout or open section in the middle of a shape, such as two overlapping circles used to create a donut. If you apply a drop shadow to the outer shape, you break the illusion of the cutout. To fix this, you can apply an inner shadow to the interior shape, adding a consistent lighting effect to the overall drawing.
- The Hide Object filter property hides the original object and shows the filter, including parts that would have been obscured if the object were visible. It's as if the object is transparent but still casting a shadow.
- Knockout works like a cookie cutter. It hides the original object but shows only the parts of the filter that would be seen if the object were visible. The filter is masked, or knocked out, by the object. Selecting Hide Object in the filter Properties panel has no effect if Knockout is also selected.

## Round-trip editing with Adobe Illustrator and Adobe Photoshop

Where Flash Catalyst falls short in the drawing department, it makes up for with its integration and round-trip editing in Illustrator and Photoshop. In this topic students learn how to launch and edit Flash Catalyst project artwork using Illustrator and Photoshop.

Students will need to have Adobe Illustrator CS5 and Adobe Photoshop CS5 installed to complete this topic. They must also have the Photoshop FXG extensions installed.

### Key skills or concepts to reinforce:

- Use Illustrator to edit bitmaps and vectors. Use Photoshop to edit bitmaps.
- To round-trip edit with Illustrator, you must have Adobe Illustrator CS5 installed.
- To round-trip edit with Photoshop, you must have Adobe Photoshop CS5 installed and have already downloaded and installed the FXG extensions. These include the FXG plug-in and the Simplify Layers For FXG script. Instructions for downloading and installing the extensions are located at www.adobe.com/go/photoshopfxg.

### Additional points for discussion or clarification:

- When you round-trip edit a vector in Illustrator and the vector is not part of a Flash Catalyst component, changes apply only in the state in which you select and edit the vector. If the vector was shared to other states, they are not updated by the edit. To update the other states, select the edited artwork (after returning it to Flash Catalyst) and choose States > Make Same In All Other States.

- You cannot edit vectors or bitmaps that you've optimized in Flash Catalyst unless you first break them apart. Select the optimized graphic and choose Modify > Break Apart Graphic.
- It's important that you make all structural changes to objects in Flash Catalyst. An example of a structural change is repositioning, adding to, or deleting objects within a group, when the group is used in multiple states or transitions. Changing the structure of objects during round-trip editing can break the indented behavior of objects and the transitions in which they occur.
- If a component is part of a mixed selection or group, the custom component appears as a non-editable placeholder layer in Photoshop or Illustrator.

## Preserving fidelity during round-trip editing

When you round-trip edit between Flash Catalyst and Illustrator or Photoshop, you are using the FXG file format to move objects between applications. Follow these tips to preserve the fidelity of your Flash Catalyst artwork and the edits you make during round-trip editing.

- The filters you add in Flash Catalyst are editable in Illustrator.
- If you rotate or apply a filter to a bitmap in Flash Catalyst, you cannot edit it in Photoshop until you rasterize the image. Choose Modify > Rasterize to do so.
- Filters and effects you add in Illustrator convert to vectors or bitmaps when returning to Flash Catalyst.
- Adding layer effects, masks, shape layers, and adjustment layers in Photoshop requires that you run the Simplify Layers for FXG script before returning to Flash Catalyst. There is no harm in running the script. A good practice is to always run the script before returning to Flash Catalyst.
- Always set Proof Colors to Monitor RGB in Illustrator to lessen the difference when comparing colors between Flash Catalyst and Illustrator. Choose View > Proof Colors (to select it), then choose View > Proof Setup > Monitor RGB.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book.*

1. What is the difference between grid lines and guide lines?
2. What causes a visible seam when overlapping two shapes with the same gradient fill?
3. Which filters, when applied in Adobe Illustrator, are editable in Flash Catalyst?

4 What does the Knockout filter property do?

5 What is required before you can round-trip edit Flash Catalyst artwork using Adobe Photoshop CS5?

6 What should you do before returning to Flash Catalyst when round-trip editing in Photoshop, if you've added an adjustment layer or mask?

## Answers

1 Using the grid is similar to placing a transparent sheet of graph paper over the artboard. The grid includes perfectly spaced vertical and horizontal lines that help you align and draw perfectly measured artwork. Unlike with the grid, you can place horizontal or vertical guides anywhere you want. Use the ruler for exact placement of your custom guides. The same guides persist across all pages in the application. The guides you set for the main pages in the application do not appear when you edit a component. When a component is in Edit-In-Place mode, you can place new guides that are unique to that component.

2 If the shapes are the same size and the gradient rotation is different, this will cause a visible seam (the gradients won't line up). If the shapes are different sizes, then you need to adjust the rotation angle of one shape until the gradients line up. In some cases this will require a little trial and error.

3 Blur, Drop Shadow, Inner Shadow, Bevel, Glow, and Inner Glow. When you import the artwork from Illustrator, make sure the Filters > Keep Editable fidelity option is selected.

4 Knockout works like a cookie cutter. It hides the original object but shows only the parts of the filter that would be seen if the object were visible. The filter is masked, or knocked out, by the object.

5 Adobe Photoshop CS5 must be installed and you must have already downloaded and installed the Photoshop FXG extensions.

6 You need to run the Simplify Layers for FXG script before returning to Flash Catalyst. There is no harm in running the script. A good practice is to always run the script before returning to Flash Catalyst.

# Lesson 12: Publishing a Project

By the time students get to Lesson 12, they have already had several opportunities to run projects in a web browser and test applications. In this lesson, students publish the application for hosting on the web, viewing locally, or distributing as an Adobe AIR application.

## Goals for this lesson

After completing this lesson, students will be able to:

- Choose a delivery option
- Prepare the application for publishing
- Publish the application files
- View and share published files
- Publish to the web using Adobe Dreamweaver CS5

## Delivery options

This topic covers the three ways to publish and distribute a project from Flash Catalyst. A fourth option is also mentioned: save the Flash Catalyst project file (FXP) and continue developing by importing the project into Adobe Flash Builder. Extending the project using Flash Builder is covered in more depth in Lesson 13.

### Key skills or concepts to reinforce:

- Using Publish to SWF/AIR in the File menu, you can publish your Flash Catalyst project for web delivery, for sharing and viewing on a local system, and for distribution as an Adobe AIR application.
- Adobe AIR is a cross-operating system runtime that allows you to build and deploy applications to the desktop. AIR applications are launched from the user's desktop and do not require an Internet connection or web browser.

### Additional points for discussion or clarification:

When a project requires extended development in Flash Builder, be sure to complete the iterative design phase in Flash Catalyst before sharing the file with a developer. There is no integration between Flash Catalyst and Flash Builder. Publishing a local version for sharing and review is a great way to get input during interaction design, before moving the project to Flash Builder.

## Publishing your application

In this topic, students publish a project using all three publish options. They output the project for the web (deploy-to-web), for local viewing (run-local), and as an Adobe AIR application.

The topic begins by explaining accessibility and font embedding, and offers tips for preparing your project before publishing. Students then publish the application.

### Key skills or concepts to reinforce:

By default all projects are built for accessibility, which makes them easily navigable by users with visual impairment (when using screen reader software). If you want your project to be accessible to screen readers, you should also make sure that you've included descriptive text for each object in the Accessible Text field of the Properties panel.

### Additional points for discussion or clarification:

- To test project accessibility, you need to have screen reader software installed. For more information on creating accessible projects, visit the Adobe Accessibility Resource Center: www.adobe.com/accessibility/?promoid=DJGVE
- The sample project that students publish includes an embeddable font. If your project doesn't include any embeddable fonts, the Embed Fonts option is disabled. For more information on embedding fonts in a Flex application, visit: http://help.adobe.com/de_DE/Flex/4.0/UsingSDK/WS2db454920e96a9e51e63e3d11c0bf69084-7f5f.html
- In step 3, make sure that students change the output directory for their published file. For this lesson, students use the same Lesson12 folder where they copied the lesson practice file. If students republish the project, they'll be asked if they want to overwrite the existing files. Have them choose Overwrite.

## Viewing the published files

Publishing a project creates a separate project folder for each type of output you choose. In the previous topic, students published all three types of output, so they generated three separate project directories. In this topic, students examine published files.

### Key skills or concepts to reinforce:

- Deploy-to-web contains the files you publish to a web server.
- Run-local contains the files you can share offline.
- AIR contains a single file used to install the AIR desktop version of the project.

### Additional points for discussion or clarification:

- You can rename the output folders to a name that is more descriptive of the application. You can also rename the AIR installation file.
- When you publish the web version of the application, make sure to upload every file and folder in the deploy-to-web folder to your server. You can rename the main.html file to something such as index.html, but do not change the name of the main.swf file, unless you also update the reference in the main.html file.
- The main.html file in the deploy-to-web folder references the swfobject.js code that performs Flash Player version checks. You can place your SWF (Flash Catalyst application) in another HTML page, but make sure that you copy the swfobject.js code into the new or existing HTML file.

- After publishing, students may try to run the application locally by opening the main.html file in the deploy-to-web folder. This version of the application must be hosted on a server to run properly.

## Publishing to the web with Adobe Dreamweaver CS5

This topic provides instructions for uploading the published deploy-to-web files to a web server using Adobe Dreamweaver CS5. The steps are specific to Dreamweaver CS5, but the process is very similar in Dreamweaver CS4.

To complete this skill, students must have Dreamweaver CS5 installed and they must have a web server to which they can publish and test the files.

If students do not have Dreamweaver CS5, but they do have an FTP client and access to a web server, you can swap this topic for instructions on publishing using the FTP client.

### Key skills or concepts to reinforce:

Publishing from Dreamweaver requires creating a new site, indicating the local root folder (use the deploy-to-web folder), connecting to the remote site, and uploading the Flash Catalyst application files.

### Additional points for discussion or clarification:

You need to provide FTP access information to students. This includes the ftp address, username, and password for logging in.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1 Which version of the application do you publish if you want to import the project into Adobe Flash Builder for further development?

2 How do you make a project accessible to screen reader software?

3 How can you test to make sure that accessibility is working?

4 Can you rename the main.swf file that's created when you publish the application?

5 Which version of the published project can you share with clients and colleagues for viewing offline?

6 What is one limitation of viewing the application locally?

## Answers

1 You don't need to publish the project to import the file into Flash Builder. Instead, save the project file (FXP). The project file (FXP) is imported directly into Flash Builder along with all of its assets.

2 First, you need to make sure that you've entered accessible text for each object in the application. You can do this by selecting objects and entering text in the Accessible Text field in the Properties panel. Then make sure that Build For Accessibility is selected in the Publish To SWF dialog box. This is selected by default.

3 The best way to check the accessibility of a project after publishing is to view the published application on a system with screen reader software installed.

4 You can rename the main.swf file, but you must also update the reference to the file in the main.html file.

5 Run-local is a larger SWF file with all of its assets included and no dependencies. The purpose of this redistributable version is to share it. For example, you can share the file with clients and they'll have everything they need to view the application locally.

6 Links to URLs (using the Go To URL interaction) don't work when viewing the run-local version of the application.

# Lesson 13: Extending Your Project Using Adobe Flash Builder

Lesson 13 explores ways that you can extend a Flash Catalyst project using Adobe Flash Builder. Projects that require binding components to data and web services are good candidates for these techniques. For example, a Data List component designed in Flash Builder can load and display records in a database at runtime. In this lesson, students are introduced to Flash Builder, including a few examples of ways to extend the functionality of an application. You revisit the designer/developer workflows that were introduced in Lesson 1, get some tips for preparing a Flash Catalyst project for further development, and then open a project in Flash Builder. Finally, students learn the steps for importing a Flash Catalyst library (FXPL) into Flash Builder.

## Goals for this lesson

After completing this lesson, students will be able to:

- Prepare the application for handoff to a developer using Flash Builder
- Import a Flash Catalyst project into Flash Builder

- Import an edited Flash Catalyst project in Flash Builder
- Compare and integrate code between two projects
- Import a Flash Catalyst project library into Flash Builder

## Adobe Flash Builder

This brief topic provides an overview of the Flash Builder integrated development environment (IDE).

### Key skills or concepts to reinforce:

- Adobe Flash Builder 4 (formerly Flex Builder) is an IDE for developing cross-platform data-centric RIAs.
- You can design entire RIAs using Flash Builder.
- You can use Flash Catalyst to design functional components (rapid prototypes or completely designed controls) and then bind those components to services in Flash Builder.

## Extending the application

In this topic, students explore some examples of ways to extend a Flash Catalyst project using Flash Builder.

### Key skills or concepts to reinforce:

- Bind components to a data source.
- Collect user input using forms.
- Implement create, read, update, and delete (CRUD) functionality.

For more information and examples of design and data-centric RIAs, visit the Adobe Customer Showcase finder: www.adobe.com/cfusion/showcase/index.cfm?event=finder&projecttypeid=16&loc=en_us

### Additional points for discussion or clarification:

This course does not cover working in Flash Builder or how to bind Flex components to services.

## Exploring designer/developer workflows

There are several possible workflows for designing RIAs using Adobe Creative Suite tools, Flash Catalyst, and Flash Builder. In this topic, students review two types of workflows: linear and iterative.

### Additional points for discussion or clarification:

Flash Catalyst and Flash Builder are not integrated, but they are designed to work together. Ideally, you should have your Flash Catalyst project as complete as possible before handing off the file to a developer. Once you import the project there is no round-trip editing capability. Most projects require iteration, and it's possible that you'll need to go back into Flash Catalyst to make changes. If this happens after work has started on the project in Flash Builder, a few techniques can help you to update your project with the most current designs in Flash Builder. These include:

- Import the edited file in Flash Builder and use code compare to identify and update code changes.
- From Flash Catalyst, export a library package. In Flash Builder, import the updated library.

## Preparing files for a developer

In this topic, students review some tips on preparing a Flash Catalyst project for import into Flash Builder.

### Key skills or concepts to reinforce:

- Meet with your development team early.
- Use design-time data in a data list.
- Name everything using descriptive names.
- Delete unused assets from the project library before saving the Flash Catalyst file or exporting a Flash Catalyst library.

## Opening a Flash Catalyst project in Flash Builder

Students begin this topic by running the Flash Catalyst project and identifying components that require completion using Flash Builder.

This topic goes as far as importing the project file (FXP) into Flash Builder, but does not include the steps to bind components or add web services.

Students must have Adobe Flash Builder installed to complete this topic. If other students have completed this topic on the same system, then you should clean up the old Flash Builder projects that are generated while completing these steps. When the students import the Flash Catalyst projects into Flash Builder, new Flash Builder projects are added. To remove them you need to:

1. Start Flash Builder.
2. Select the project in the Package Explorer, and choose Edit > Delete > Yes.
3. Delete the project folder at the location specified when the FXP file was imported.

**Key skills or concepts to reinforce:**

- When you import the FXP file, the entire project, including all project assets and artwork are added to the project and available in Flash Builder.
- To view and run the project, open the Main.mxml file in the [*project name*]/src/default package/Main.mxml file. Then choose Run > Run Main.

**Additional points for discussion or clarification:**

- This topic includes the steps for importing an edited Flash Catalyst project and then comparing and integrating code between the old and new projects. This process is a work-around for the lack of integration between Flash Catalyst and Flash Builder. The better approach is to get final sign off on the Flash Catalyst design before moving into Flash Builder development.
- When importing an FXP project created with Adobe Flash Catalyst, the imported project can contain references to fonts that are not available on your system. The Import wizard provides the option to fix font references using CSS. If you select this option (it's selected by default), Flash Builder imports the Flash Catalyst style sheet Main.css. Main.css contains references to the fonts used in the project.

## Importing a Flash Catalyst library file (FXPL)

In this final topic of the course, students follow the steps to import a Flash Catalyst Library Package (FXPL) into Flash Builder. This creates a new Flex library project in Flash Builder. The Flash Catalyst project assets can then be used in one or more Flex applications in Flash Builder.

**Additional points for discussion or clarification:**

If you end up editing the components or creating new components in Flash Catalyst, export the new FXPL from Flash Catalyst. Then import the FXPL into Flash Builder and follow the steps in "Associate the FXPL with your application" to replace the existing Flex Library with the new one. For this approach to work, avoid making changes to files within the components package in Flash Builder.

## Questions

The following questions are not in the student's *Adobe Flash Catalyst CS5 Classroom in a Book*.

1. In addition to opening an FXP file in Flash Builder, what is another way to make Flash Catalyst project assets available in a Flex project?
2. Can you perform round-trip editing between Flash Catalyst and Flash Builder?
3. When importing an FXP into Flash Builder, what happens if you select the option to Fix Font References In CSS?

4 When you import a Flash Catalyst project file into Flash Builder, where can you go in Flash Builder to view and edit the MXML code that was generated by Flash Catalyst?

5 If you want to compare and update code after making changes to the repeated item template in a Data List component, which file would you need to update?

6 Which Flash Builder project file do you open when you want to define calls to remote operations that will fetch data at runtime?

## Answers

1 Export the Flash Catalyst project library as an FXPL file. Then open the FXPL in Flash Builder and associate the library with your Flex application.

2 No. In version 1.0 of Flash Catalyst CS5, there is no integration between Flash Catalyst and Flash Builder. It's best to finalize the interaction design before opening the FXP file in Flash Builder. If you do make changes, you can attempt to update the Flex project using code compare, or by importing an updated library and associating the new library with your Flex project.

3 When importing an FXP project created with Adobe Flash Catalyst, the imported project can contain references to fonts that are not available on your system. The Import wizard provides the option to fix font references using CSS. If you select this option, Flash Builder imports the Flash Catalyst style sheet Main.css. Main.css contains references to the fonts used in the project.

4 Open the Main.mxml project file, which is stored in [*project name*]/src/default package/Main.mxml file. Each component is defined in a separate MXML file. The component files are stored in [*project name*]/src/components.

5 To compare and update the code in the repeated item templates for a data list, you need to locate the MXML file for that repeated item. This will be stored in the [*project name*]/src/components directory. When using the code compare feature in Flash Builder, this file should be identified automatically when Flash Builder compares the code between the two projects.

6 In Flash Builder, open the Main.mxml file and use the Data/Services panel to connect the application to a data service.