

Creating ActionScript 3.0 components in Flash CS3 Professional – Part 9: Shim compiled clip

Jeff Kameron

Adobe

Welcome to the last part of the article series on creating components using ActionScript 3.0. In this section we'll optimize our MenuBar component and prepare it for distribution. We'll also discuss potential naming compatibility issues and provide some guidance on how to make components accessible for screen readers. If you missed the articles leading up to this last part of the series, I recommend going back and starting at the beginning. In Part 1 of the series you'll find a link where you can download the sample files for the entire series. Or if you'd like to just follow along with this section of the series, download the sample files for this section in the Requirements section below.

Creating a shim compiled clip

As I described in Part 5 of this article series, the ComponentShim holds precompiled ActionScript definitions for the User Component Infrastructure. A compiled clip like this enables much faster publishes of FLA files that contain components in them and also allows you to distribute components without distributing the source code. If you want to know more about the ComponentShim compiled clip and how it works, be sure to read the sidebar titled the Mysterious ComponentShim.

You can create a shim compiled clip for your ActionScript definitions. Below I'll provide instructions on how to remove the ComponentShim from your component's Library and use your shim instead. It's important to note that your shim should never be named ComponentShim, because it may run into name conflicts with the ComponentShim used by the User Interface components.

In the case of the MenuBar component, I named my compiled clip the MenuBarShim. The first step was to set up MenuBarShim.fla, which I used to create the MenuBarShim.

Here are the steps to follow:

1. Create a new ActionScript 3.0 FLA file
2. Save the file as MenuBarShim.fla, in the same directory as MenuBar.fla
3. Click the Library panel context menu and select New Symbol...
4. In the Symbol Properties dialog box:
 - a. Enter "MenuBarShim source" for the name
 - b. Select movie clip as the type
 - c. Check the option to Export for ActionScript
 - d. Change the class name to "fl.example.MenuBarShim"
 - e. Click OK

- f. You'll be presented with the ActionScript Class Warning dialog box, which says that a definition of the class will be automatically generated
 - g. Click OK
5. Click the Library panel context menu and select New Symbol... again
 - a. Enter "MenuBar" for the name
 - b. Select movie clip as the type
 - c. Check the option to Export for ActionScript
 - d. Change the class name to "fl.example.MenuBar"
 - e. Leave the option to Export on first frame checked
 - f. Click OK
6. Open MenuBar.fla
7. Drag ComponentShim from the Library of MenuBar.fla into the Library of MenuBarShim.fla
8. Right-click ComponentShim in the Library panel and select Linkage... from the context menu
9. In the Linkage dialog box, check the option to Export on first frame
10. Click OK.
11. Save MenuBarShim.fla

Compatibility issues between ComponentShim and your shim

When generating your compiled clip shim, it is very important that the definitions for User Interface Component Infrastructure definitions come from the ComponentShim used by the User Interface components. The definitions should not be compiled by adding the ActionScript source for the User Interface components to the classpath.

When Flash developer has added both your shim and ComponentShim to their FLA file, there will be some duplicate definitions in these two compiled clips. The ActionScript 3.0 compiler looks at time stamps embedded within the compiled clips to determine which definitions should be used; by default the compiler uses the definitions with the most recent time stamp.

By compiling your shim from ComponentShim, you ensure that the time stamps on all compiled definitions in MenuBarShim will be exactly the same as they are within ComponentShim, and that exactly the same versions of the precompiled code will be used. If you compiled from source, it would be much easier to accidentally use a customized version of a User Interface Component Infrastructure class, which could cause potential problems when Flash developers are using both your component and the User Interface components in their projects.

Compiling from source could also change the time stamp data, which in some cases could cause the wrong version of a definition to be used. This could happen if the Flash developer had an updated version of ComponentShim in their project. If the time stamps in your shim, which had the old definitions of the Infrastructure classes, turned out to be

more recent than those in an updated ComponentShim, bug fixes that were implemented might be ignored. This situation could even cause the updated User Interface components to break completely.

If Adobe releases an update to the User Interface components and changes ComponentShim in the future, you should retest your component with the updated User Interface Component Infrastructure. At that time, it may be prudent to regenerate your shim using the updated ComponentShim.

At this point in the process, MenuBarShim.fla was all set up and I was ready to create the MenuBarShim. To create the initial MenuBarShim and set it up in MenuBar.fla, follow these steps:

- Open MenuBarShim.fla
- Right-click the Library symbol's MenuBarShim source and select the option to Convert to Compiled Clip from the context menu
- Rename the resulting compiled clip from MenuBarShim source SWF to MenuBarShim
- Open the Linkage dialog box for the MenuBarShim compiled clip and uncheck the option to Export on first frame
- Drag MenuBarShim into the Component Assets/_private folder in the Library of MenuBar.fla
- Delete ComponentShim from the Component Assets/_private folder in the Library of MenuBar.fla
- Close MenuBarShim.fla without saving the changes
- In MenuBar.fla, edit the MenuBar symbol
- Lock and hide all layers with the exception of the componentshim layer
- Drag MenuBarShim from the Library onto Frame 2 of the componentshim layer
- Using the Property inspector, set the X and Y location of the MenuBarShim instance to 10.0 and 10.0
- Show all layers. As a best practice, you should never leave layers hidden in your component movie clip, because the option to Export hidden layers in the publish setting could be unchecked in the component user's FLA file
- Lock all layers with the exception of the assets layer
- Collapse the _private and Component Assets folders in the Library
- Save MenuBar.fla

There's one thing that deserves some explanation. You may find it a little confusing that while MenuBarShim is an SWC-based component, I never actually exported an SWC file. There is a good reason for this. I could have exported MenuBarShim.swc into the Components directory, selected Reload from the context menu of the Components panel and then dragged MenuBarShim into MenuBar.fla from the Components panel. However, when I deployed MenuBar.fla in the Components directory, I would have seen MenuBarShim listed next to MenuBar in the Components panel. Exposing MenuBarShim in this way would obviously confuse the users of your component. Selecting the option to Convert to Compiled Clip is, in most respects, equivalent to

selecting the option to Export SWC File... with a few important differences. In this case the most important difference is that it keeps the shim name from appearing in the Components panel.

I wanted to test MenuBarShim and make sure that it was working properly. MenuBar.fla has MenuBar.as and the other MenuBar source files in its classpath, so if I selected Control > Test Movie from MenuBar.fla, I knew it would not test MenuBarShim. Up until this point I had kept test.fla next to MenuBar.fla specifically so that the source files would be in its classpath. However, for testing purposes, I needed to ensure that the source files were no longer in the classpath, so I created a subfolder named test and moved test.fla into it. Since I had removed ComponentShim from MenuBar and wanted to get that update, I first deleted the entire Component Assets folder from test.fla. Then I dragged MenuBar from the Library of MenuBar.fla into the Library of test.fla to get the updated version with MenuBarShim. I dragged the Button component from the Components panel into my Library to refresh all of the Button assets that had been deleted from the Component Assets folder. Finally, I selected Control > Test Movie from test.fla, and it worked! Just to be sure the file was really using MenuBarShim, I saved test.fla, deleted MenuBarShim from the Library and selected Control > Test Movie again. This time compile errors were generated from my frame script, which is exactly what I expected. I closed test.fla without saving the changes to restore MenuBarShim to the Library.

As I was developing the component, it was necessary to recreate MenuBarShim.fla every time I made code changes. While developing and testing code changes, I would compile from source. To compile from source I could either move the test.fla file back to its original directory next to MenuBar.fla, or I could add .. to the classpath of test.fla, or I could even copy some of the source files into the test directory with test.fla. For more details on this topic, read the sidebar entitled Compiling from Source vs. From ComponentShim.

To recreate the MenuBarShim and replace it in MenuBar.fla, follow these steps:

- Open MenuBarShim.fla
- Right-click the Library symbol MenuBarShim source and select the option to Convert to Compiled Clip from the context menu
- Rename the resulting compiled clip from MenuBarShim source SWF to MenuBarShim
- Open the Linkage dialog box for the MenuBarShim compiled clip and uncheck the option to Export on first frame
- Drag MenuBarShim into the Component Assets/_private folder in the Library of MenuBar.fla
- In the Resolve Library Conflict dialog box, select the option to Replace existing items and click OK. If you do not see this dialog box, it means that you did not drag MenuBarShim into the _private folder correctly
- Close MenuBarShim.fla without saving the changes

Accessibility

As I was working on this project, I started to add screen reader support to the MenuBar component. At first I was really scratching my head on what updates to make for accessibility and how to implement them. But then I looked at the source code for the Adobe Flex 2 MenuBar and Menu components, which helped a lot. I found it extremely helpful to analyze the existing component code, because without examining it, I would never have figured out the correct messages to dispatch without doing a lot of research on my own.

After some deliberation, I decided not to add accessibility as a part of this series. I realized that if you really want to include accessibility functionality in the components you develop, you'll need to do research and really dig into the source for the User Interface components or for the Flex 2 components. The process will be different, depending on the component you are creating. So if you need to add screen reader support, then I recommend finding a component in Adobe Flash CS3 or in the Adobe Flex 2 component sets that is similar to yours, then use its accessibility code as your guide when you make the changes.

The good news for developers is that the accessibility architecture for the two component sets is virtually identical. If you are looking at Flex components, refer to the classes in the `mx.accessibility` package. If you are looking at Flash components, look at the classes in the `fl.accessibility` package. Your component will need to define a static `createAccessibilityImplementation` variable and will also need to override the method `initializeAccessibility()`.

To find more resources and information about accessibility, visit the [Accessibility Resource Center](#).

Where to go from here

All the documentation for the ActionScript 3.0 Flash Player APIs and the Adobe Flash CS3 component APIs is available in the Help panel, but I find that the three frame format of the documentation available online is much easier to navigate. You'll find the ActionScript 3.0 Language and Components Reference at <http://livedocs.adobe.com/flash/9.0/ActionScriptLangRefV3/>. A bookmark to this page is indispensable for any Flash programmer!

The source files themselves are also an invaluable resource. Sometimes you will just need to search through the code and read the code, but you will also find it very helpful to add the code to your FLA file's classpath and debug the code using the ActionScript

3.0 debugger. The User Interface Component Infrastructure source is installed with Flash CS3 at the following location:

Windows	Macintosh
C:\Program Files\Adobe\Adobe Flash CS3\language\Configuration\Component Source\ActionScript 3.0\User Interface\	/Applications/Adobe Flash CS3/Configuration/Component Source\ActionScript 3.0/User Interface/

Also remember that while the Adobe Flex 2 components are based on a framework that is different in important ways, the documentation and the source code of these components can also be useful references. The Adobe Flex 2 Language Reference is available online, also in the great three frame format, at <http://livedocs.adobe.com/flex/2/langref/>. The Flex SDK includes the complete source to the Flex component set. If you do not already have the Flex SDK installed, you can download it from [Flex.org](http://flex.org).

About the author

Jeff Kameron is a computer scientist at Adobe Systems who has worked on the Flash authoring team since 2002.