

# Building run time shared libraries

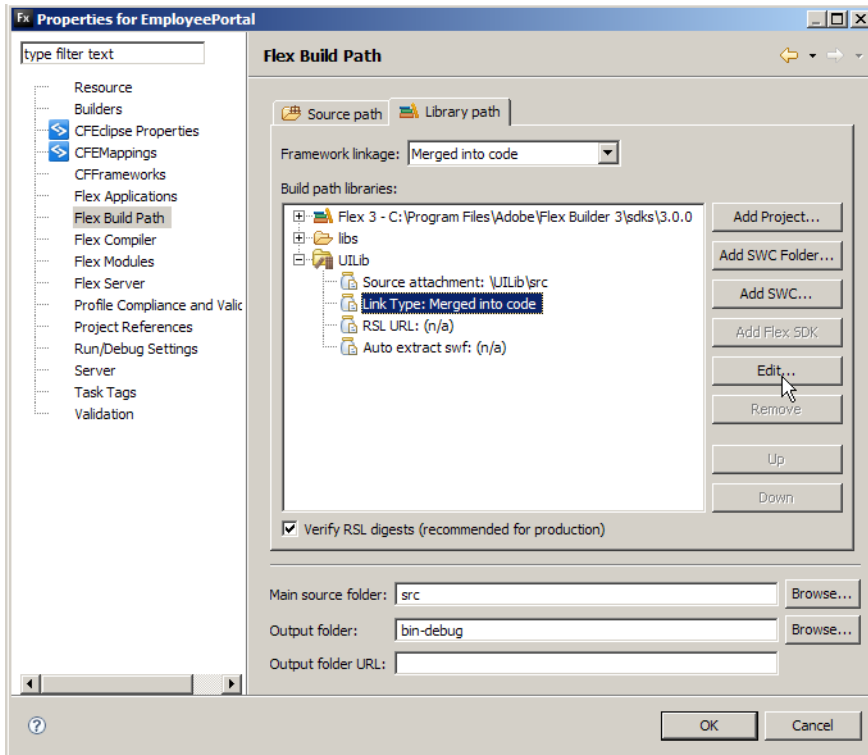
---

## Import the Project

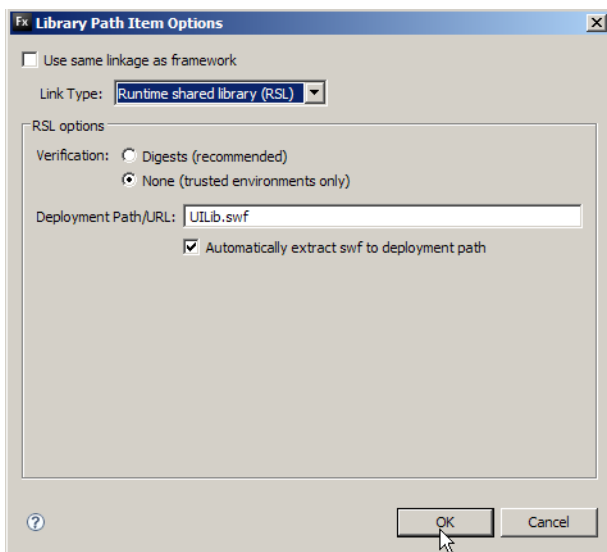
1. If not already created, create a directory named `adobeFlexTraining` on your **C** drive.
2. In Flex Builder, delete the **EmployeePortal** project if created previously. If you don't delete the contents the files will be overwritten with the new project files.
3. In Flex Builder, select **File > Import > Flex Project**.
4. In the dialog window, select **Archive File** and browse to where `Ex22_Starter.zip` is located in your local file system.
5. Uncheck **Use default location**.
6. Enter `C:\adobeFlexTraining\EmployeePortal`.
7. Click **Finish**.

## File size of the swf file in the imported project

8. Using the **Flex Navigator** view, open the **EmployeePortal > bin-debug** folder.
9. Select the `EmployeePortal.swf` file and right-click on it.
10. Select the **Properties** and notice the **Size**. It should be about 310 KB.
11. Click **OK**.
12. Right-click on the project name **EmployeePortal** and select **Properties**.
13. Select **Flex Build Path** from the left hand menu.
14. Select the **Library path** tab.
15. Notice the **UILib** exists as one of the **Build Path Libraries** associated with the project.
16. Expand **UILib** and Select **Link Type** and click on the **Edit** tab.



17. Select **Run time shared library (RSL)** from the **Link Type** menu drop down.
18. Click **OK**.



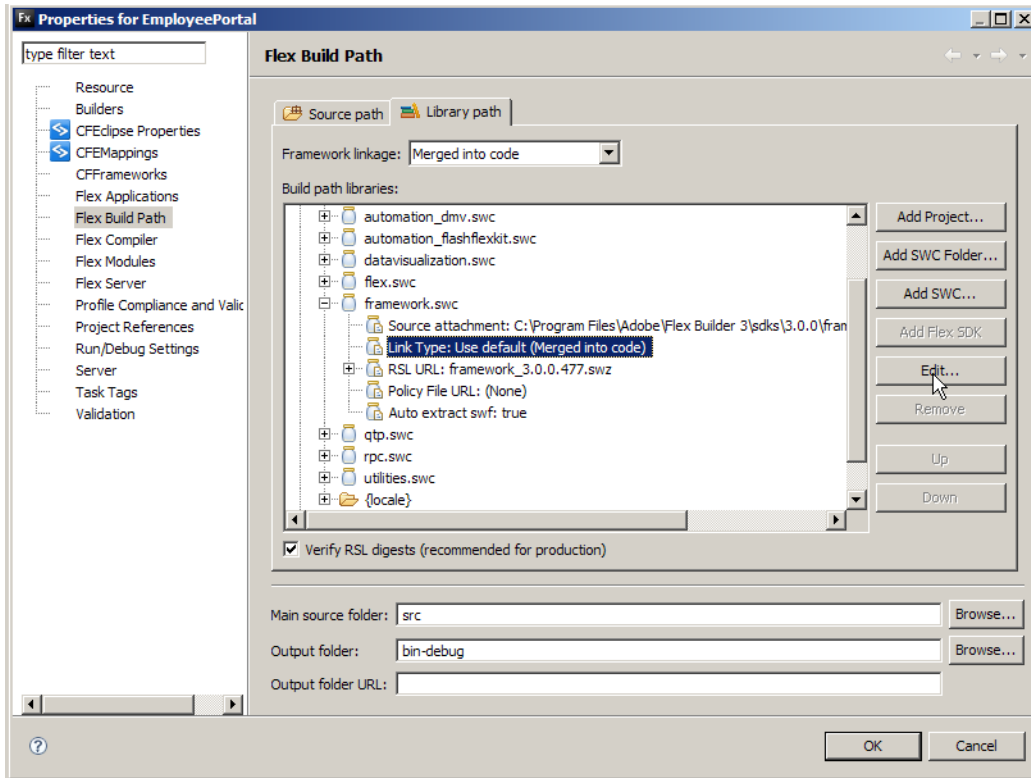
19. Click **OK** to close the **Properties for EmployeePortal** dialog window.

This step changes the application settings to link the `UILib` library at runtime instead of compiling it with the code of the application. As a result, the swf file created for the application becomes smaller in size since it does not bundle the library within it.

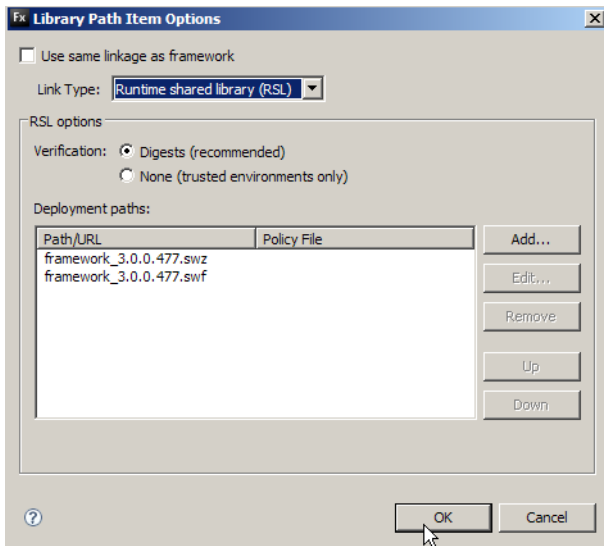
20. Select the `EmployeePortal.swf` file and right-click on it.
21. Select the **Properties** and look at the **Size** of the file. It should be about 124KB, which is a significant decrease in size compared to before.
22. Click **OK**.
23. Right-click on the project name **EmployeePortal** and select **Properties**.
24. Select **Flex Build Path** from the left hand menu.
25. Select the **Library path** tab and notice the **Flex 3.x** as one of the libraries in the project.
26. Expand the **Flex 3.x** library.

Besides the `UILib.swc` file which we created and linked to the project, there are core Flex compiler files needed for every application. One of them is the `framework.swc` file which can be seen within the Flex library. This swc file is one of the largest in the Flex Builder and is used by all application. If we change the link type of the `framework.swc` to RSL, the application will request to download this file only **once** at runtime. The file is then stored in the browser cache. For any subsequent applications using the `framework.swc` file, the flash player automatically used the copy saved in the browser cache and does not download it again. This increases the efficiency of the applications.

27. Expand **framework.swc**.
28. Select the **Link Type** and click on the **Edit** tab.



29. Uncheck **Use same linkage as framework**.
30. Change the **Link Type** menu drop down to **Run time shared library (RSL)**.
31. Click **OK**.

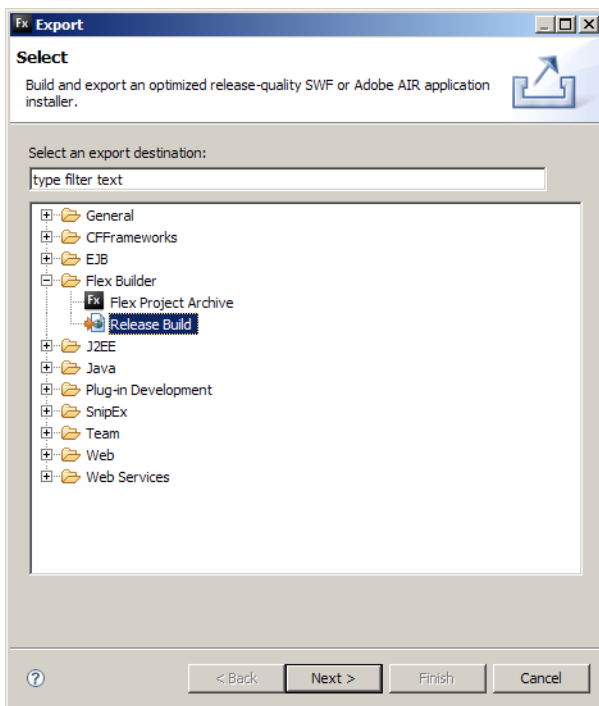


32. Click **OK** to close the **Properties for EmployeePortal** dialog window.
33. Select the `EmployeePortal.swf` file and right-click on it.

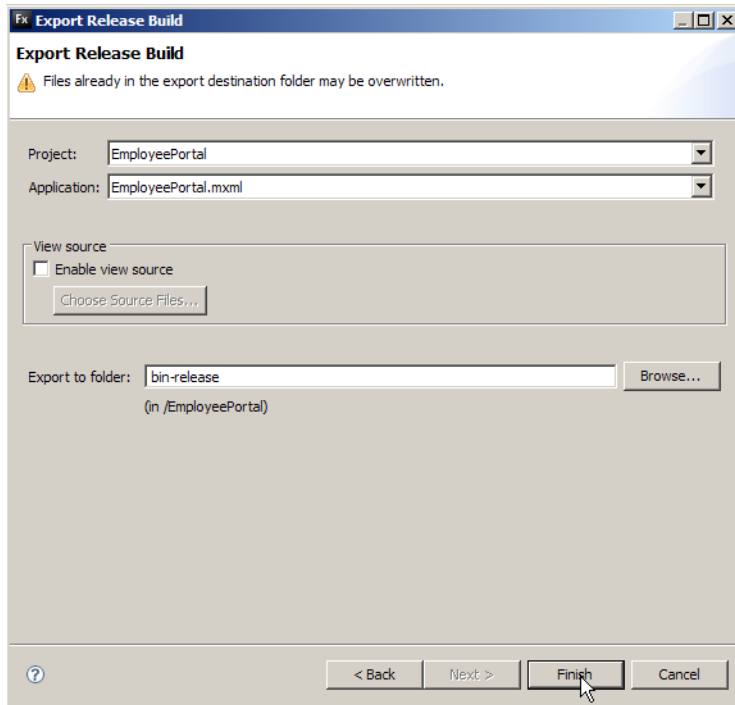
34. Select the **Properties** and look at the **Size** of the file. The file size should be about 83 KB, which is a further reduced from before.

By default, Flex Builder created the swf files for the projects under the bin-debug folder. All files under this folder have some additional debug information which is not required when publishing a project. We will create a release version of this application next.

35. Click **OK** to close the EmployeePortal.swf **Properties**.
36. Right-click on the **EmployeePortal** project and select **Export**.
37. Expand the **Flex Builder** folder.



38. Select **Release Build**
  39. Click **Next**.
- Alternatively, you can select from the main menu **Project > Export Release Build**.
40. In the dialog window that comes up, notice the **Export to folder** option is listed as **bin-release**.

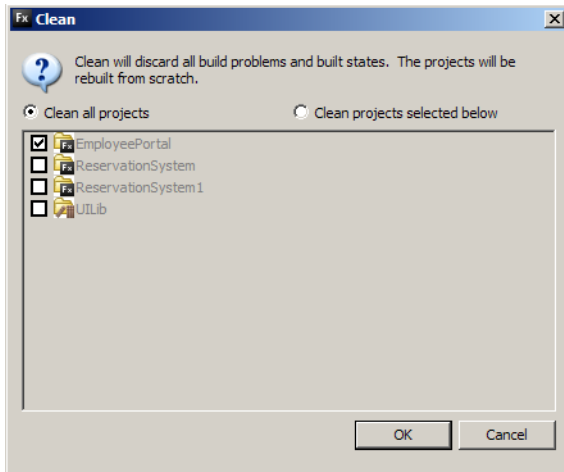


41. Click on **Finish** to create the release files.
42. Notice the **bin-release** folder is created within the **EmployeePortal** project.
43. Select the `EmployeePortal.swf` file located within the **bin-release** project and right-click on it.
44. Select the **Properties** and look at the **Size** of the file. The file size should be about 56 KB, which is a much smaller file than the original 310 KB.
45. Click **OK**.

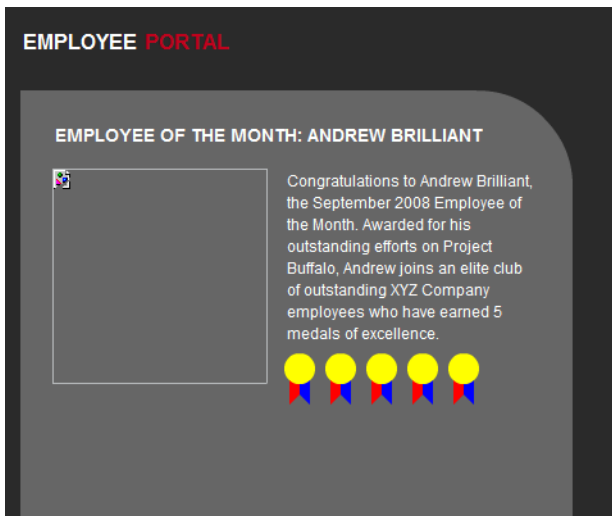
## Moving the images to the Flex Library Project

Runtime shared libraries do not directly allow us to dynamically link anything other than classes. That means we cannot directly add a dynamic link to an asset such as an image, a sound, or a font. However, if we embed an asset, we can add indirect dynamic linking. The following steps illustrate how to achieve indirect dynamic linking with images.

46. Move the images folder under **EmployeePortal > src** to **UILib > src**.
47. Clean the project by selecting **Project > Clean** for **EmployeePortal**.



48. Run the `EmployeePortal.mxml` file and notice the image of the employee does not come up anymore.



49. Using the **Flex Navigator** view, open the **UILib > src > components** folder.
50. Open the `EOM.mxml` file and locate the `<mx:Image>` tag.
51. Change the code to have the `@Embed` tag in the source property around the path to the employee image.

Your code should appear as follows.

```
<mx:Image source="@Embed('images/abrilliant_large.jpg')"
width="167" height="167" />
```

52. Save and run the `EmployeePortal.mxml` file again and notice the image of the employee is restored.

## Package names within the swc files

When linking an external swc file or a Flex library project to an application, it may sometimes be confusing to know the package structure within the swc file. In the absence of an API that helps the developer to know which files are in which directories, here is a tip to get the correct namespace for your project.

53. Open the `EmployeePortal.xml` file, and delete the **components** namespace in the `<mx:Application>` tag on the top. The exact code that you will delete is `xmlns:components="components.*"`
54. Before the `<mx:Script>` block, start writing the code for a Form tag. (`<Form>`)

Before you complete the tag for the form, you will notice a drop down with the list of components available. By choosing the tag starting with the word **components**, Flex Builder will add the namespace with the correct directory or package structure to the application.

You may delete the Form tag.