Profiling OpenLaszlo Applications with Flash Builder

Created on: 07/4/2012
Last updated on: 07/4/2012
Version: 1.0
Author: Raju Bitter / rajubitter@googlemail.com

- Profiling OpenLaszlo Applications with Flash Builder
  - Overview
    - Software Requirements and Setup
    - Installing Tomcat and OpenLaszlo
    - Creating the Flash Builder Project
      - Create ActionScript project: Step 1
      - Create ActionScript project: Step 2
      - Create ActionScript project: Step 3
      - Create ActionScript project: Step 4
  - Debugging and Profiling
    - Development Workflow
    - Profiling the Application
    - Debugging the OpenLaszlo LFC classes
    - Further Questions and Bug Reports

Overview

This document explains how to debug and profile OpenLaszlo applications using Flash Builder. OpenLaszlo is a rich Internet application platform with a declarative UI definition language called LZX. LZX files can be compiled into either JavaScript or Adobe Flash SWF files. For SWF files, an additional compile step firs compiles LZX into ActionScript 3, and then the embedded Adobe Flex (for future versions Apache Flex) compiler is utilized to compile the ActionScript 3 source into an SWF file.

The generated ActionScript 3 code can be debugged and profiled using the existing IDEs and development tools of the Flash ecosystem (e.g. Flash Builder, Flash Develop, etc.). This document describes a possible setup of Apache Tomcat with the OpenLaszlo server, the Eclipse based Flash Builder IDE and additional tools to enable a comfortable workflow when debugging and profiling ActionScript 3 code generated by the OpenLaszlo server.

Software Requirements and Setup

The following software is required for this tutorial:

- OS X or Windows operating system (Adobe has dropped support for Flash Builder and Linux)
- Oracle Java 6 SE SDK (32 bit)
- Apache Tomcat server

- Apache Ant
- Adobe Flash Builder 4.6
- OpenLaszlo server (LPS) with SWF11 runtime support (currently only available as nightly build of the flex4.6 branch)
- Tools for generating ActionScript 3 source code as well as the LFC SWF11 runtime source code: openlaszlo-flashbuilder-project on GitHub

Install the Java SDK, and set the $JAVA_HOME environment variable. Install Apache Ant, and add both $JAVA_HOME/bin and the $ANT_HOME/bin folder to your PATH. Download and install Flash Builder.

Create a folder as a work space for the OpenLaszlo web application and the project files, e.g

```
C:\work
```

Installing Tomcat and OpenLaszlo

Download a nightly build of the OpenLaszlo flex4.6 branch. Unzip the file, and put the content into a subfolder . You should have the following folder structure.

```
C:\work
└───openlaszlo
    ├───lps
    └───WEB-INF
```

Install Apache Tomcat, and deploy the folder containing the OpenLaszlo server as a web application. The simplest way to deploy the folder is to create a new context XML file named openlaszlo.xml in

$TOMCAT_HOME/conf/Catalina/localhost

Here is an example context file with the OpenLaszlo server deployed in folder `C:\work\openlaszlo`:

```
<Context path="/openlaszlo" docBase="c:\work\openlaszlo" />
```

Download the Zip file with the example project setup here. Unzip the content of the file into the folder profiling with the LPS. Here is the resulting folder structure on my machine:

```
C:\work
    └───openlaszlo
        ├───lps
        ├───profiling
        │   ├───docs
        │   ├───html-template
        │   ├───jars
        │   ├───lfc-src
        │   ├───lzx
        │   ├───src
        │   └───swc
        └───WEB-INF
```

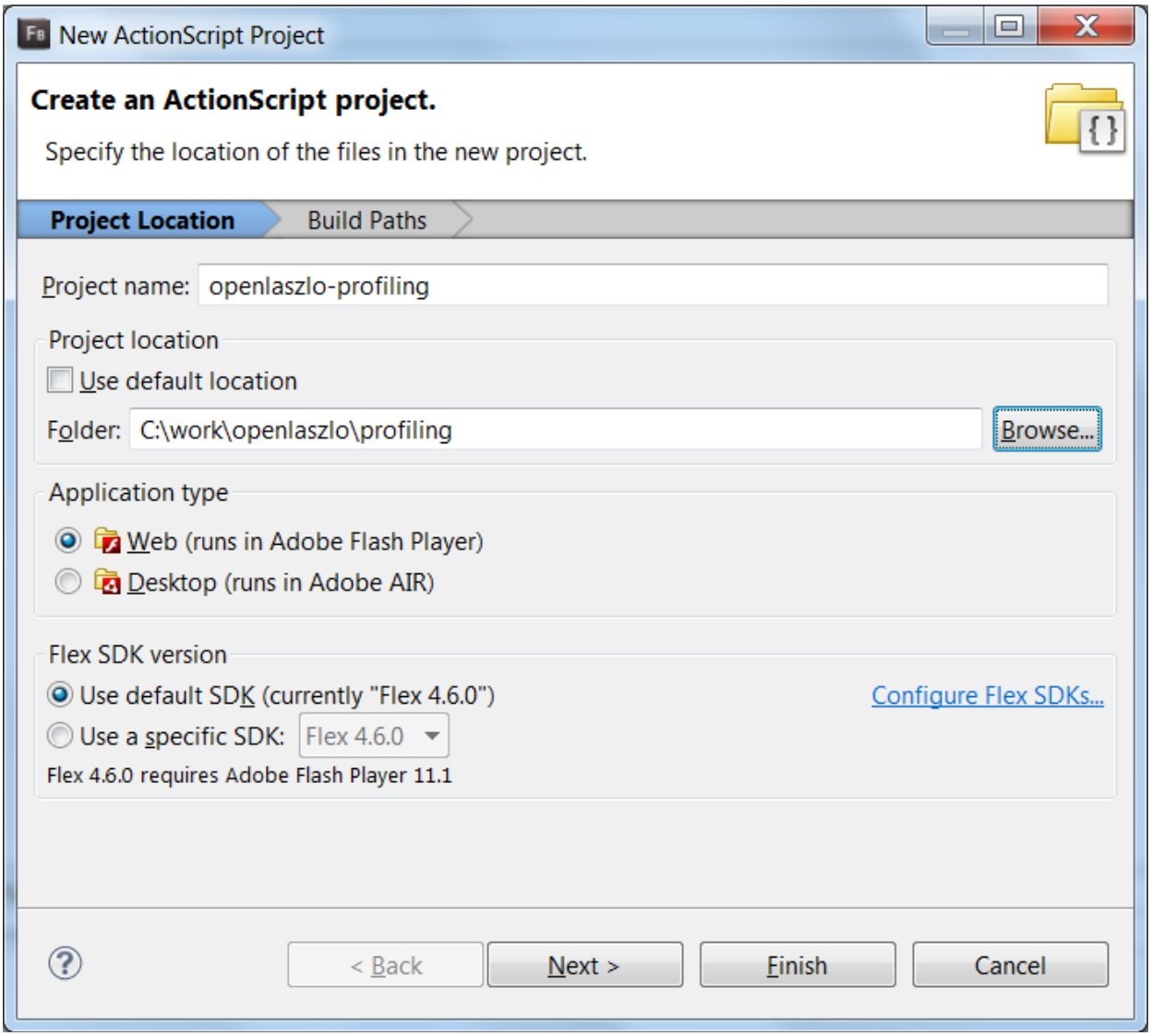
The LZX application main file is in profiling/lzx/main.lzx. I can compile that application using the LPS and Tomcat using the URL http://localhost:8080/openlaszlo/profiling/lzx/main.lzx.

Creating the Flash Builder Project

The next step is to create a Flash Builder ActionScript project. Launch Flash Builder, and create a new ActionScript Project:

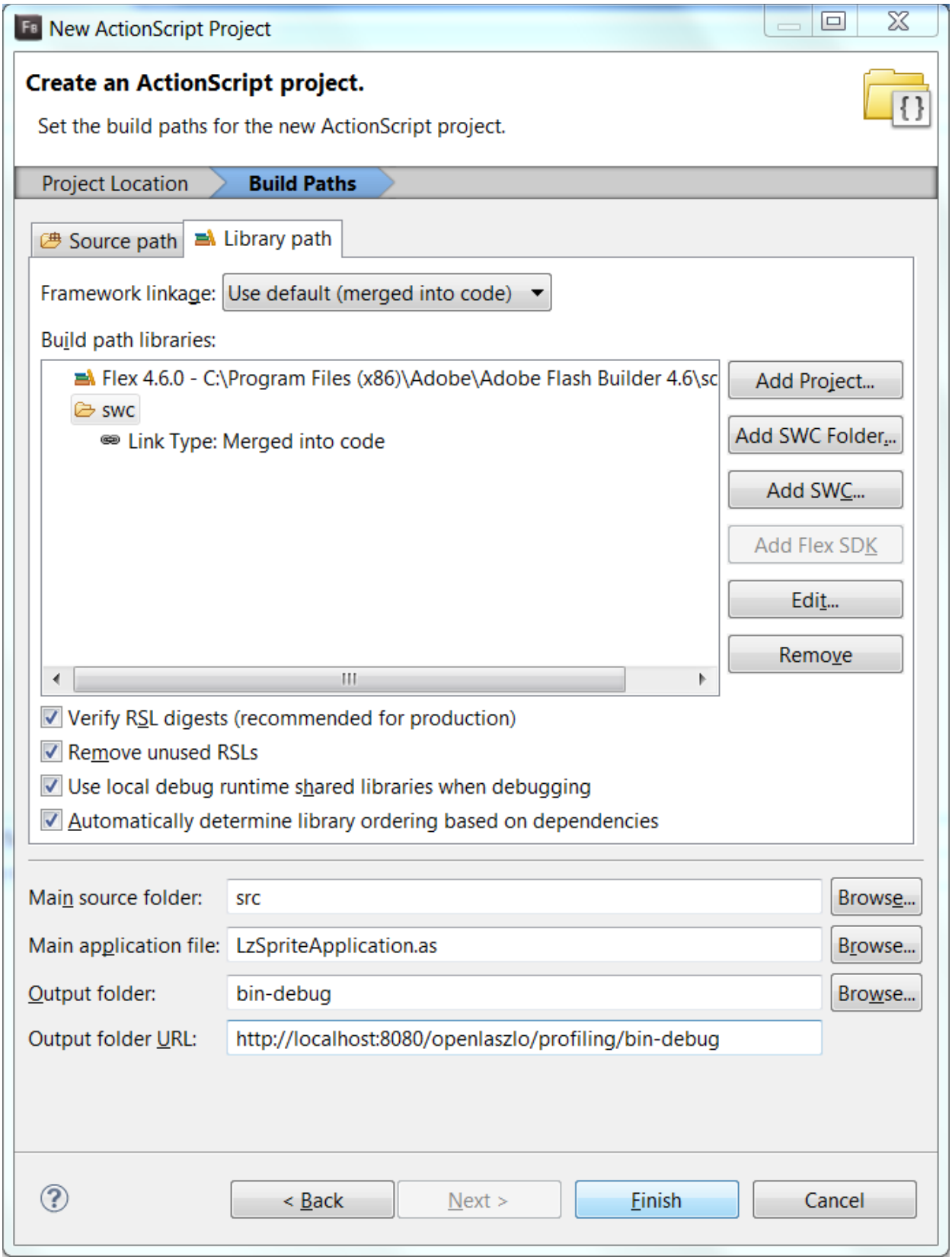
Create ActionScript project: Step 1

1. Set the project name, e.g. *openlaszlo-profiling*.
2. Select the project location folder which contains the project template, e.g. `C:\work\openlaszlo\profiling`
3. Click next.



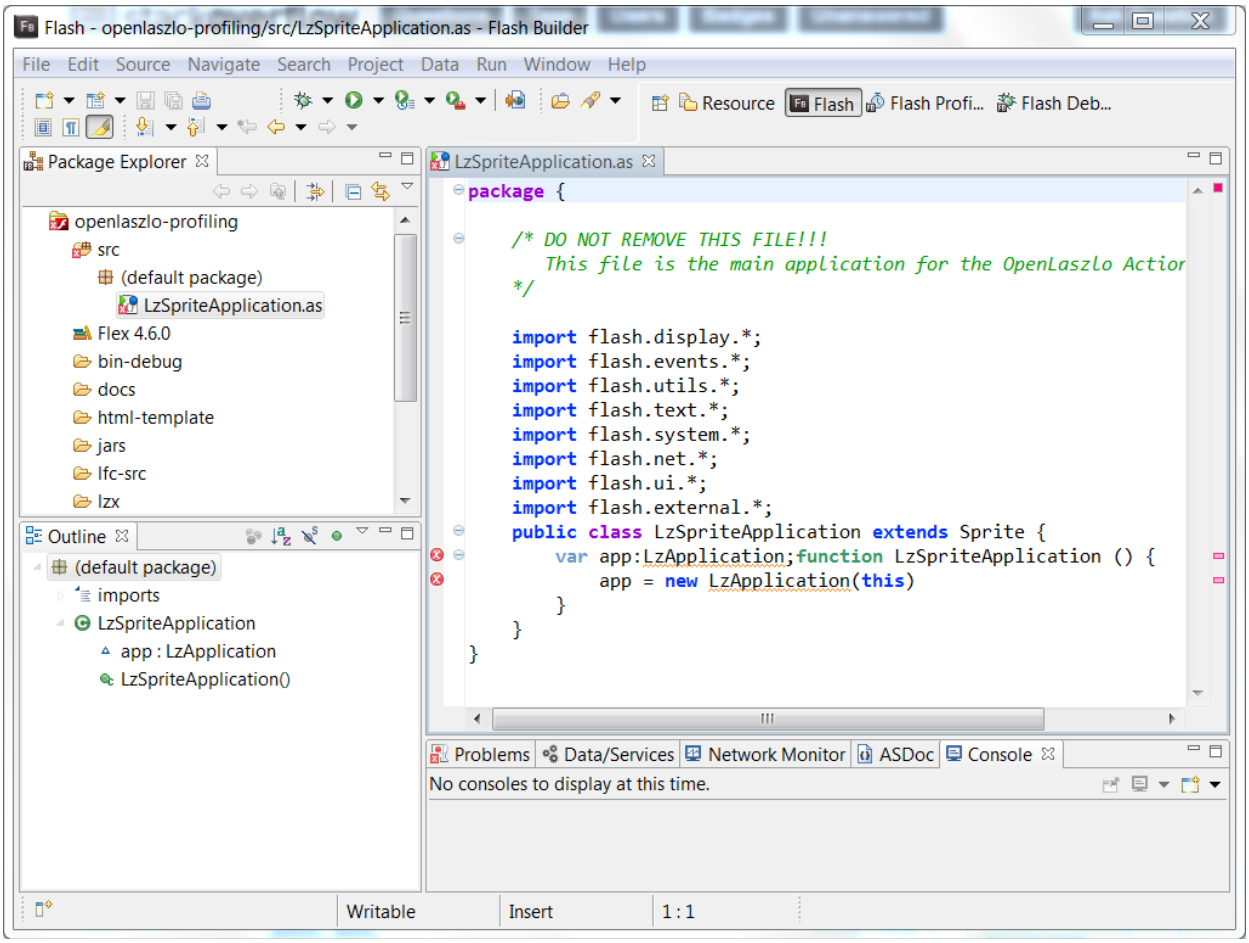
Create ActionScript project: Step 2

1. Add the folder `profiling/swc` as an SWC folder. Add any 3rd party SWCs for your application to this folder.
2. Select LzSpriteApplication.as as the main application file.
3. Set the output folder URL to http://localhost:8080/openlaszlo/profiling/bin-debug/
4. Click *Finish*.



Create ActionScript project: Step 3

You should now see the Flash Builder project with the LzSpriteApplication.as file open. The compiler will complain about the missing LzApplication class. Once we have the Ant build script running, those error messages will disappear.

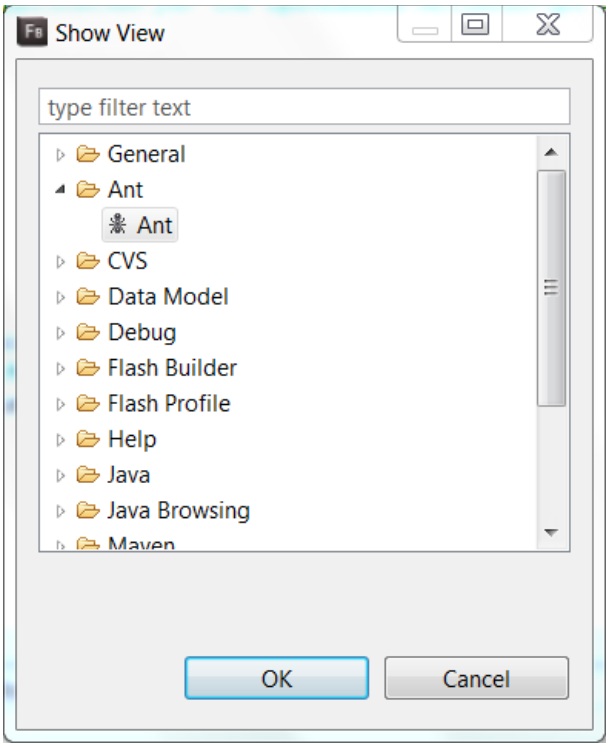


Create ActionScript project: Step 4

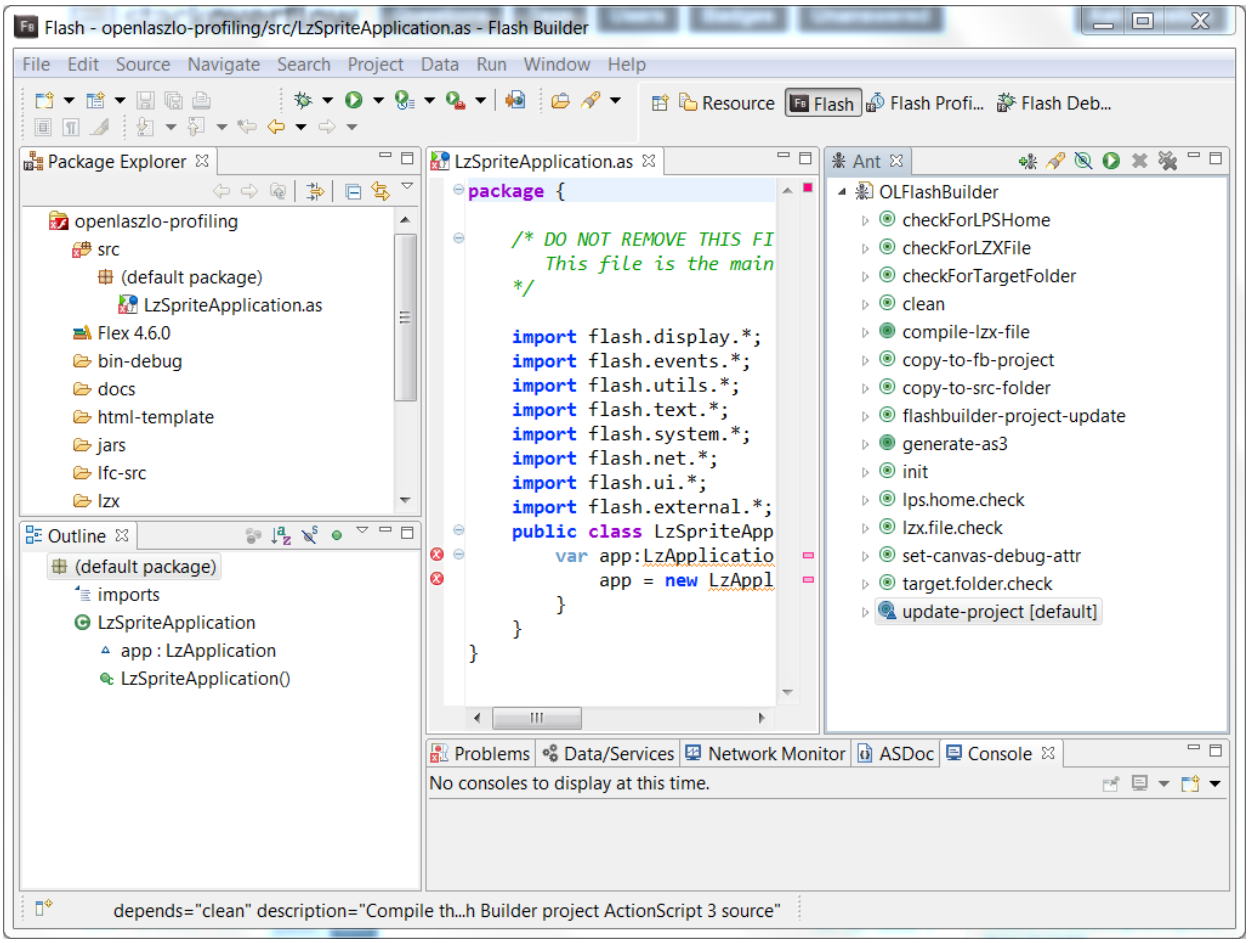
Open the build.properties file and set the LPS_HOME to point to your LPS home folder, e.g.

```
# OpenLaszlo LPS_HOME folder: has to point to a nightly build of
OpenLaszlo flex4.6 branch
LPS_HOME=C:/work/openlaszlo
```

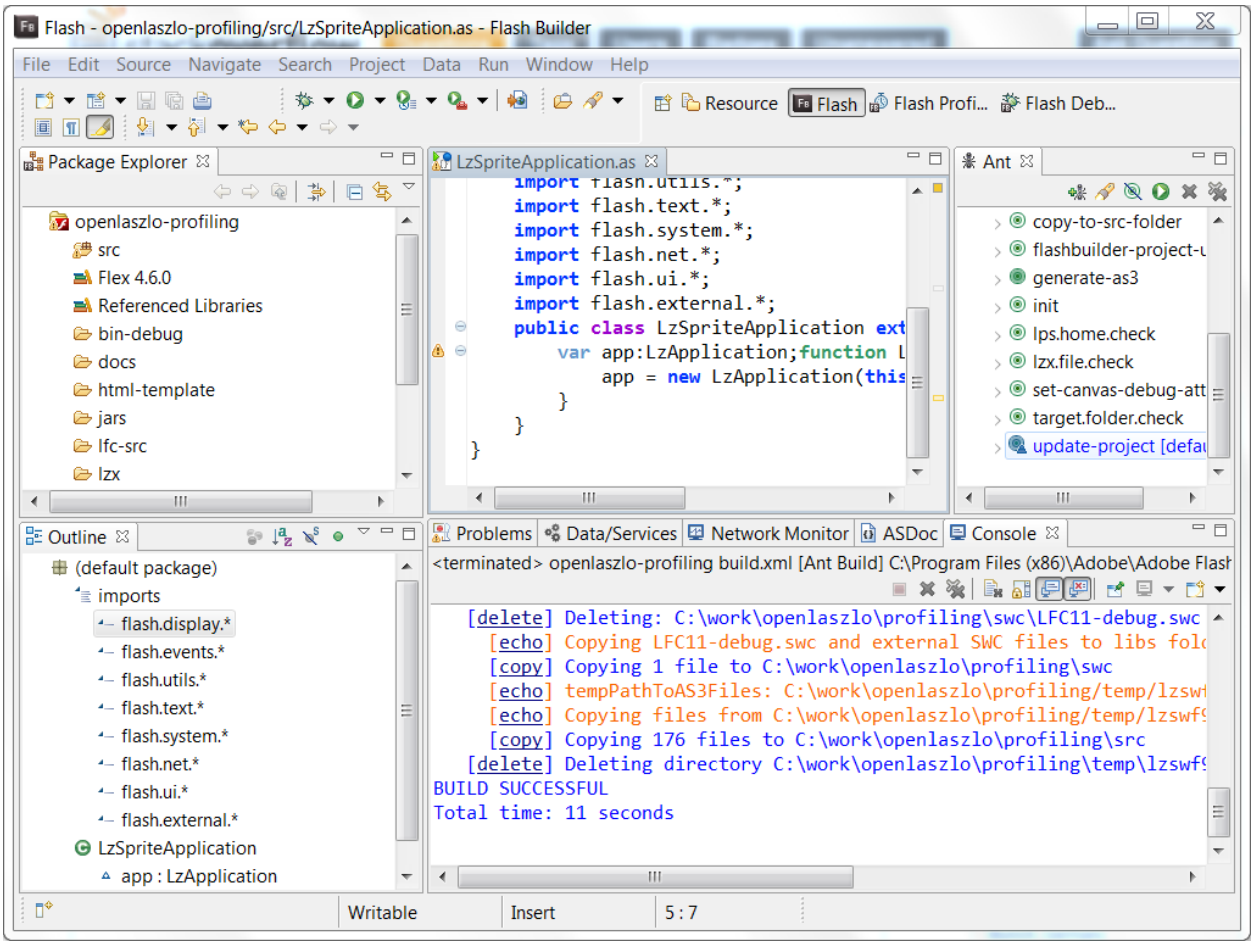
Add the Ant view in Eclipse to your Flash perspective (Window -> Show View -> Other):



Add a build file to the view (either by dragging build.xml from the package explorer into the view, or by clicking the *Add Buildfiles* button.



Run the default target by clicking on the green play icon. If everything is setup correctly, the build process should start. You should see some output in the console, but no error messages. Once the build process has run through, the error messages in the LzSpriteApplication.as file should go away. You are now ready to debug and profile your OpenLaszlo application.



Debugging and Profiling

Development Workflow

With this project setup, you can continue to modify the LZX code and compile it as you would normally do, using the LPS server and the URL

http://localhost:8080/openlaszlo/profiling/lzx/main.lzx

Once you need to debug or profile your application in the SWF11 runtime, just run the Ant build script to generate the ActionScript code and launch the debugger or profiler out of Flash Builder.

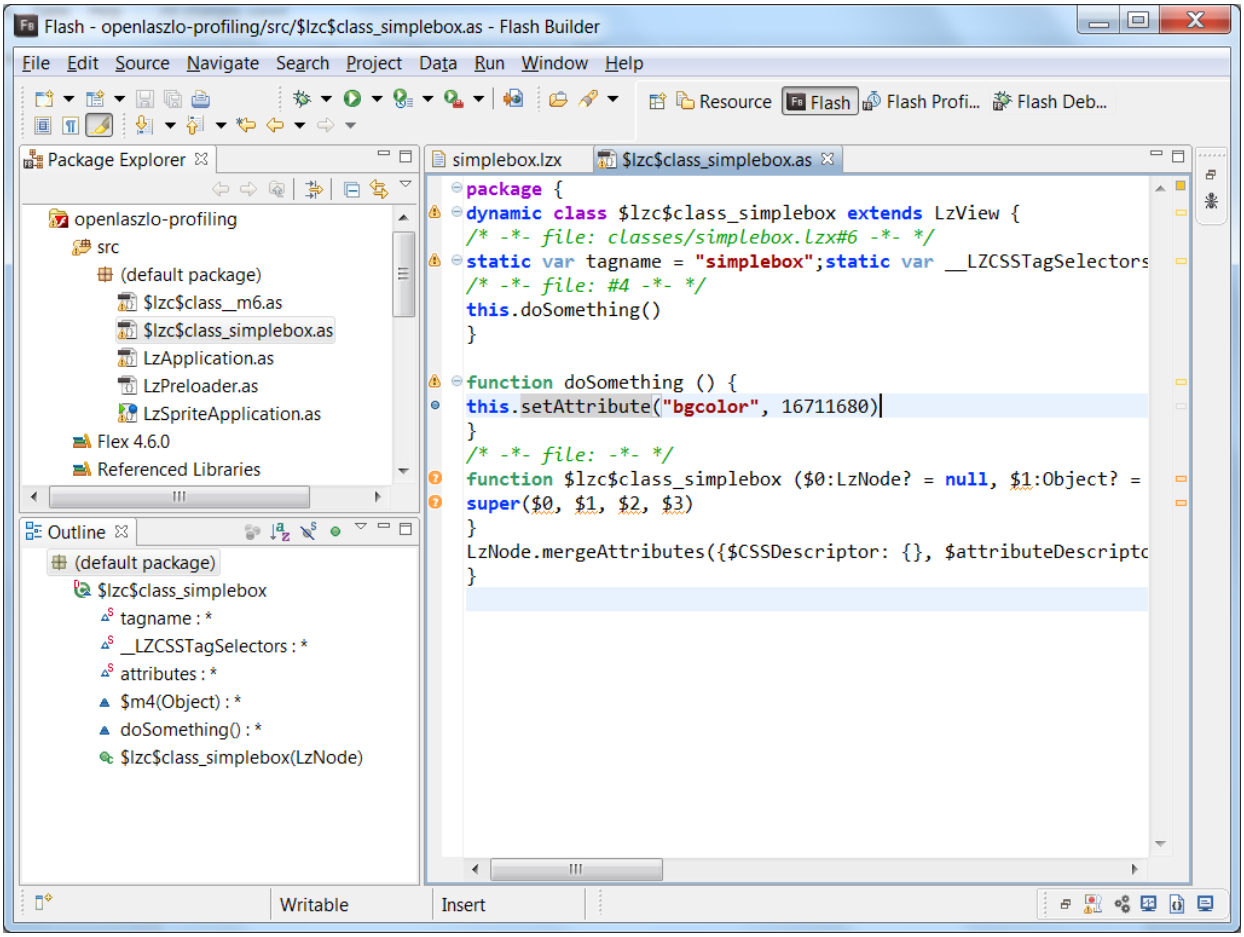
Let's start with modifying an LZX file, `src/lzx/classes/simplebox.lzx`. We will add a method doSomething, which will set the simplebox background color to red when clicked.

```
<class name="simplebox" extends="view"
       onclick="this.doSomething()">
     <attribute name="width" value="250" />
     <attribute name="height" value="250" />
```

```
        <method name="doSomething">
             this.setAttribute('bgcolor', #ff0000);
        </method>
</class>
```

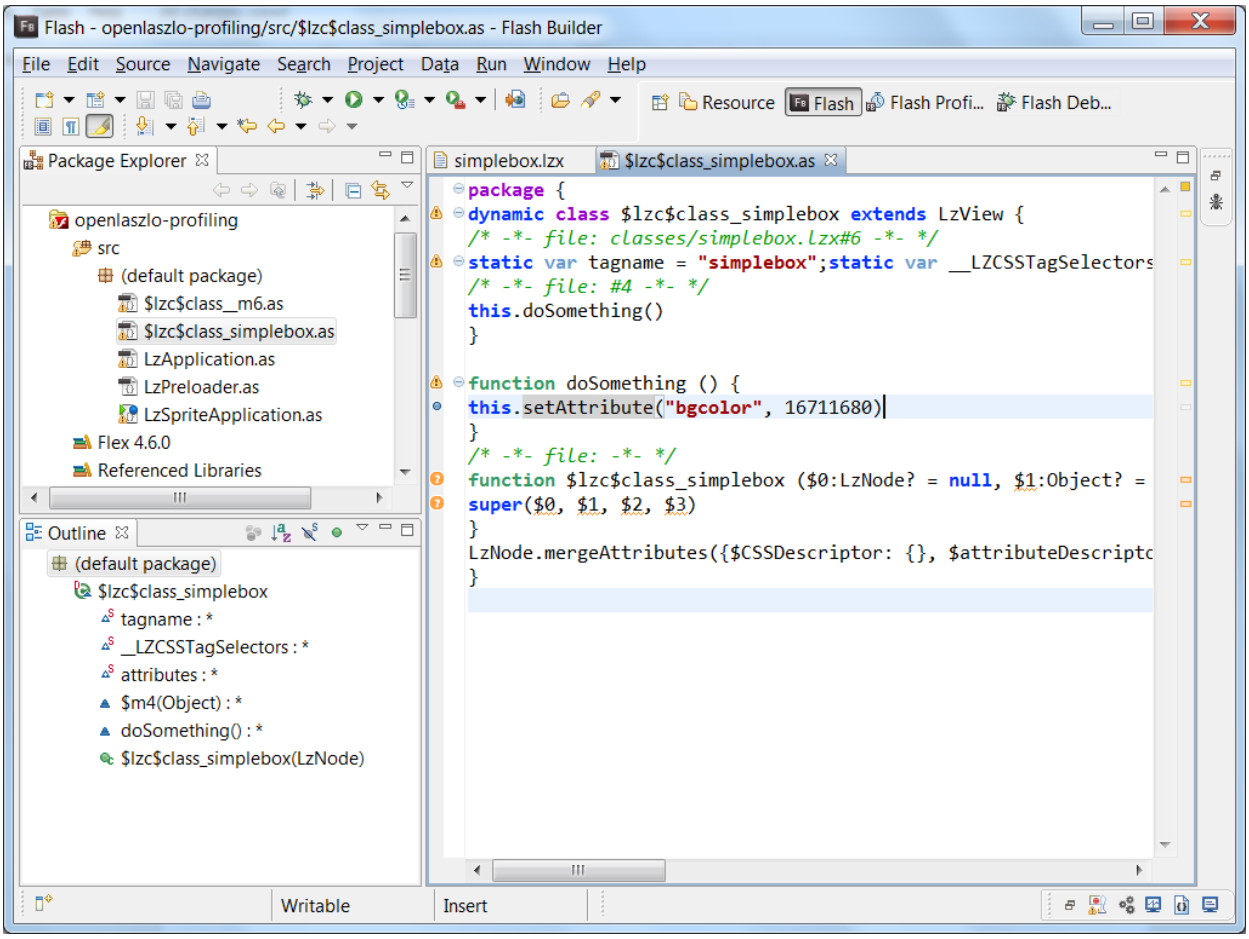
Compile the application using the URL http://localhost:8080/openlaszlo/profiling/lzx/main.lzx. Click on the transparent blue view, and the background color will be set to red.

Now launch the Ant build script. Look into the src folder, and you will see a class file with the name `$lzc$class_simplebox.as`. The OpenLaszlo compiler will automatically add the `$lzc$class_` prefix to any class generated out of LZX code.

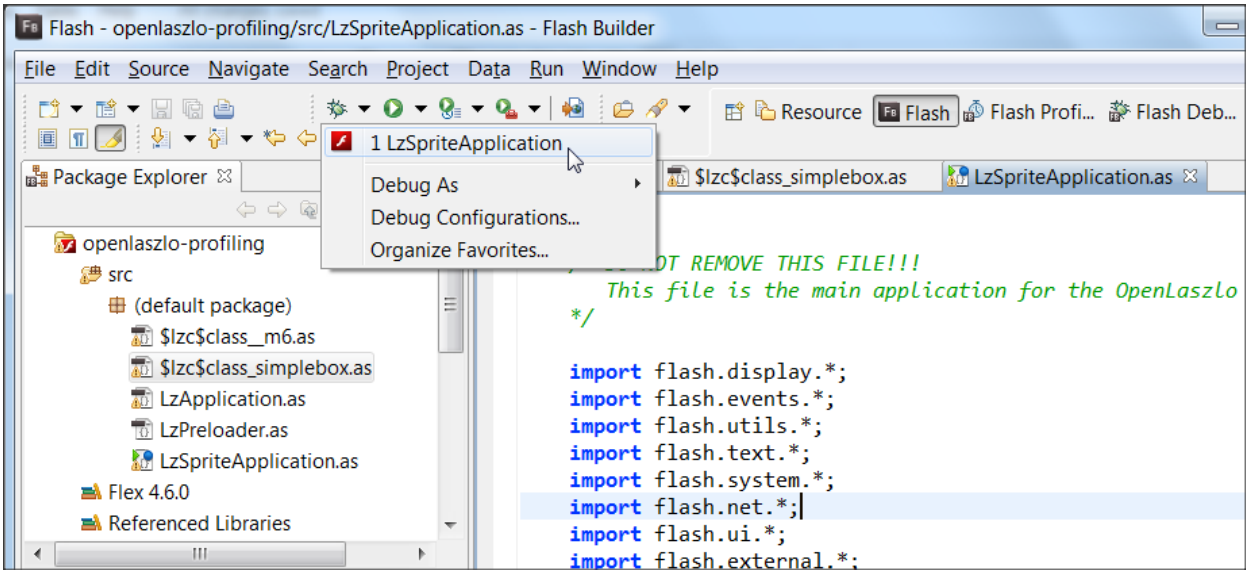


The class has a function doSomething():

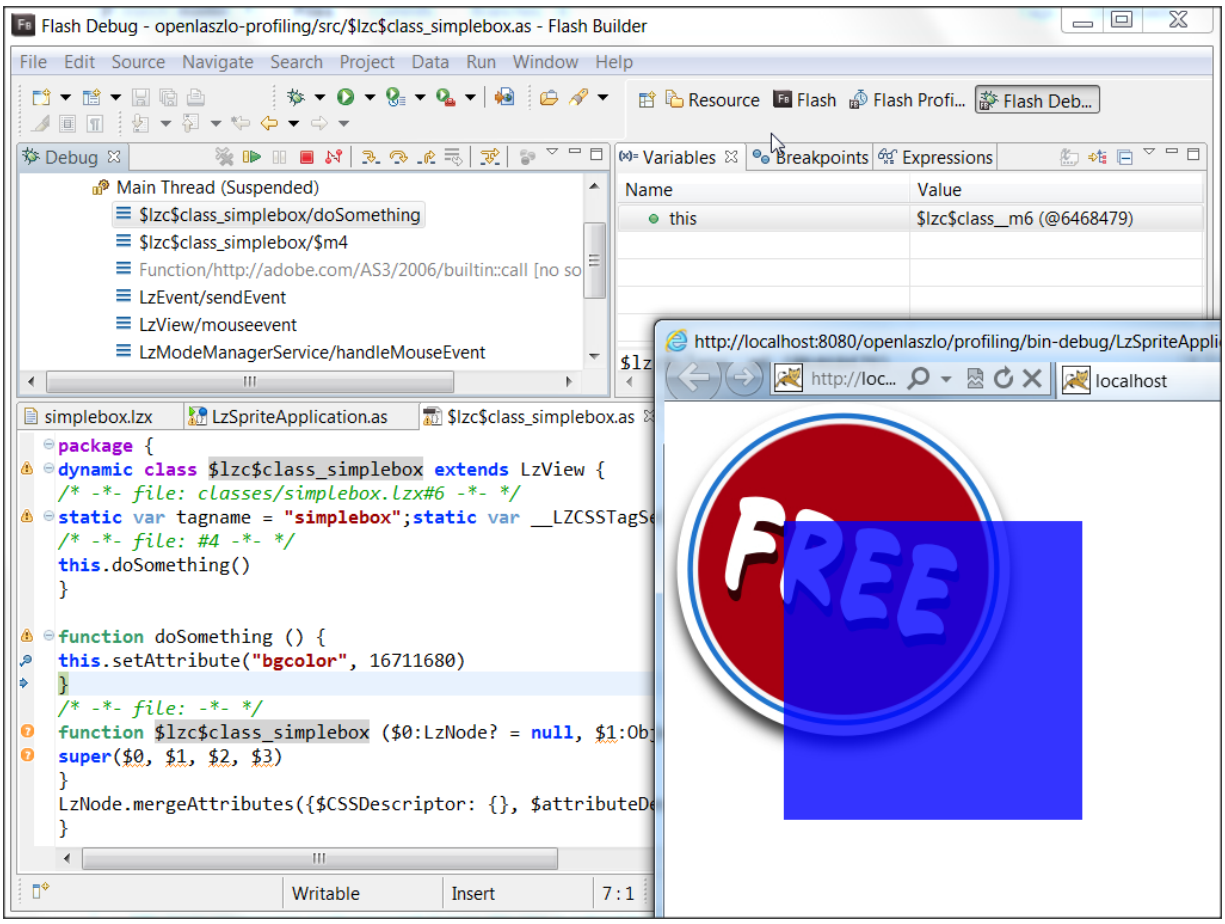
```
function doSomething () {
this.setAttribute("bgcolor", 16711680)
}
```



Set a breakpoint in the line where the bgcolor is set by clicking in the grey area at the left side of the editor view. You should see a blue circle once the breakpoint is set. Now open the `LzSpriteApplication` file and debug the application.

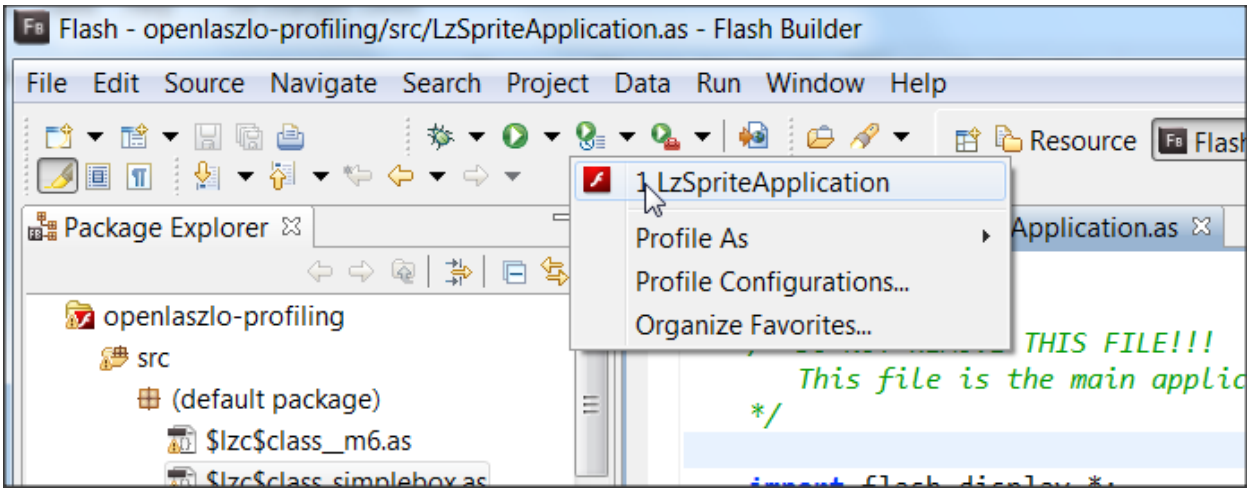


The browser will open with the application running, and once you click on the blue view, the Flash Builder will ask you to open the debug perspective of the IDE. The IDE will show the line with the breakpoint highlighted, and you can now inspect the variables and step through the code.



Profiling the Application

To profile the application, open the `LzSpriteApplication.as` file and click the profile button.



The Profiling Perspective will be opened, click and *resume* and you can profile your application. Read the section Using the Profiler of the Flash Builder documentation to learn more about profiling ActionScript applications.

Debugging the OpenLaszlo LFC classes

Debugging and stepping through OpenLaszlo's LFC classes only makes sense if you have a basic understanding of the LFC structure and good knowledge of ActionScript 3. The original source code for the ActionScript LFC is not bundled with the OpenLaszlo server, but only available through the Subversion repository. The original code can be found here:

http://svn.openlaszlo.org/openlaszlo/branches/flex4.6/WEB-INF/lps/lfc/

The LFC contains files written in a JavaScript dialect called LaszloScript, and a runtime kernel written in ActionScript directly.

http://svn.openlaszlo.org/openlaszlo/branches/flex4.6/WEB-INF/lps/lfc/kernel/swf9/

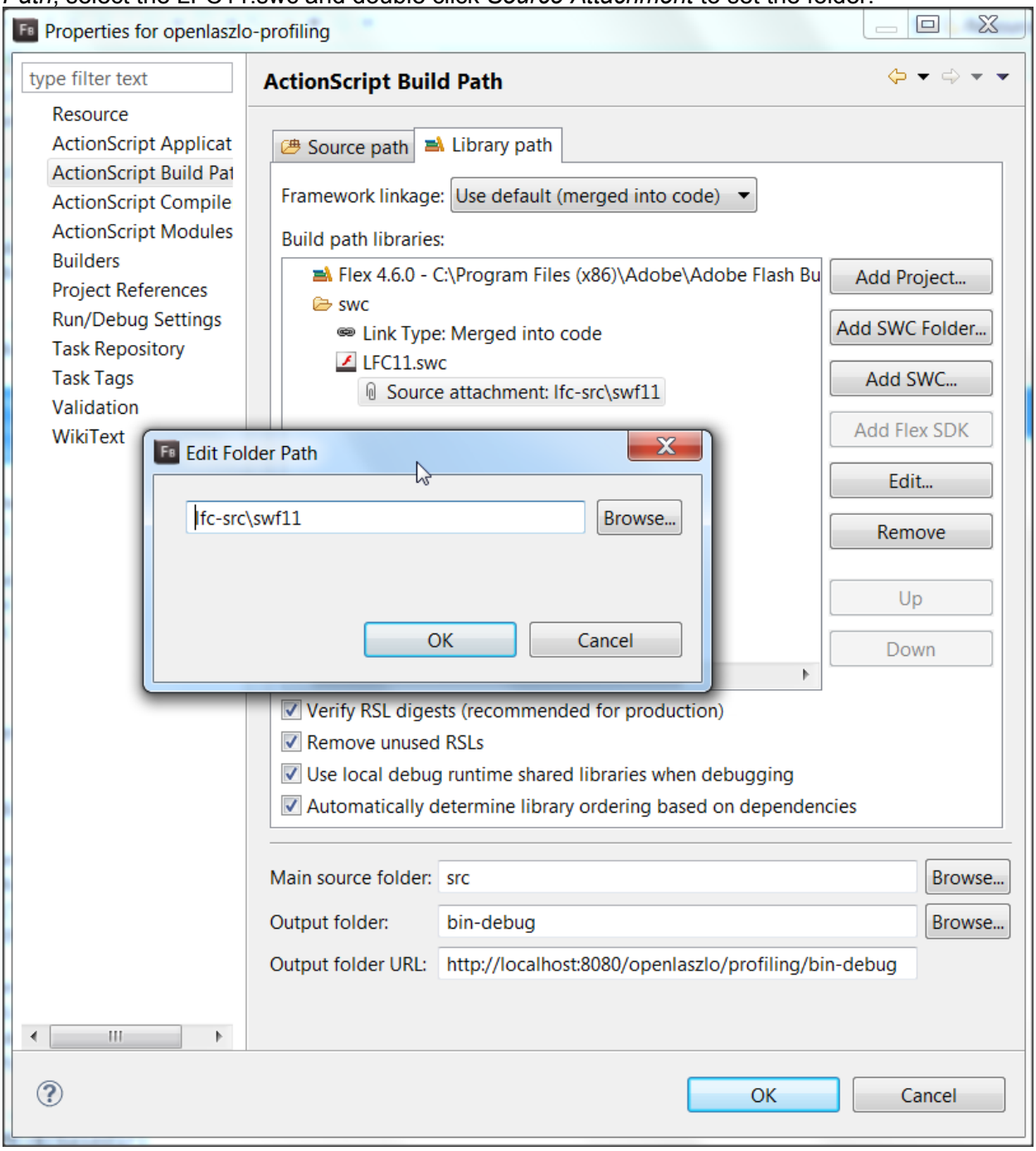
During the OpenLaszlo server build process, the LaszloScript files are compiled into ActionScript 3, and then these files as well as the kernel ActionScript classes are compiled into an SWC library.

I have generated the ActionScript classes for the SWF11 LFC using the Subversion build process, unfortunately there is no other way to have access to the LFC source files at the moment. Therefore debugging of LFC classes using this project setup is currently only possible for SWF11 runtime without the OpenLaszlo debugger enabled.

To debug the LFC classes, make sure that the ActionScript 3 source code is generated with debug disabled. The debug mode is controlled by a setting in the build.properties file:

```
# Debug settings
# OpenLaszlo canvas @debug
LZX_DEBUG_FLAG=false
```

The pre-configured project contains the source code for the non-debug SWF11 runtime LFC (flex4.6 branch, revision 19709). With the LZX_DEBUG_FLAG set to *false*, run the Ant build script. This will copy the LFC11.swc into the swc folder. Open the project preferences, and configure the source code attachment for the LFC11.swc library. The source code for the SWC is available in folder `lfc-src/swf11`. Open the *project properties*, select *ActionScript Build Path*, select the LFC11.swc and double click *Source Attachment* to set the folder.



You can now set breakpoints in any of the ActionScript classes of the LFC and step through the LFC code as well as your application code.

Further Questions and Bug Reports

If you have any questions or something does not work for you, please report any problems in the OpenLaszlo community forums at Assembla.com.