JavaFX Scene Builder

Using JavaFX Scene Builder with Java IDEs

Release 2.0

E51283-01

April 2014

This document gives information about how to use JavaFX Scene Builder with the NetBeans, Eclipse, or IntelliJ IDEs.

JavaFX Scene Builder Using JavaFX Scene Builder with Java IDEs Release 2.0

E51283-01

Primary Author: Cindy Castillo

Contributor: Jerome Cambon

Contents

Preface

This preface gives an overview about this tutorial and also describes the document accessibility features and conventions used in this tutorial - *Using JavaFX Scene Builder with Java IDEs*.

About This Tutorial

This document gives information about how to use JavaFX Scene Builder with the NetBeans, Eclipse, or IntelliJ IDEs.

Audience

This document is intended for JavaFX developers.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the following documents in the JavaFX Scene Builder and JavaFX documentation sets:

- *JavaFX Scene Builder Installation Guide*
- *JavaFX Scene Builder Release Notes*
- *Getting Started with JavaFX Scene Builder*
- *JavaFX Scene Builder User Guide*
- *Mastering FXML*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

This chapter gives a brief introduction about JavaFX Scene Builder and gives a brief introduction on what is covered in this document

JavaFX Scene Builder (Scene Builder) enables you to quickly design JavaFX application user interfaces by dragging a UI component from a library of UI components and dropping it into a content view area. The FXML code for the UI layout that you create in the tool is automatically generated in the background. To learn more about Scene Builder's features, see JavaFX Scene Builder User Guide.

Scene Builder can be used as a standalone design tool, but it can also be used in conjunction with Java IDEs so that you can use the IDE to write, build, and run the controller source code that you use with your application's user interface. Although Scene Builder is more tightly integrated with NetBeans IDE, it is also integrated with other Java IDEs described in this document. The integration enables you to open an FXML document using Scene Builder, run the Scene Builder samples, and generate a template for the controller source file.

The following chapters give information about how to configure the NetBeans, Eclipse, or IntelliJ IDEs to use with Scene Builder:

- Using Scene Builder with NetBeans IDE
- Using Scene Builder with Login Eclipse IDE
- Using Scene Builder with IntelliJ IDEA

2

Using Scene Builder with NetBeans IDE

This chapter describes how to download and install NetBeans IDE, use it to create a new JavaFX FXML project, start Scene Builder from within the IDE, and run Scene Builder sample applications.

The integration of JavaFX Scene Builder with NetBeans IDE provides optimal development workflow. It is assumed that you have already installed Scene Builder before continuing with the rest of this chapter. Go to the JavaFX Scene Builder Installation Guide for installation information.

Downloading and Installing NetBeans IDE

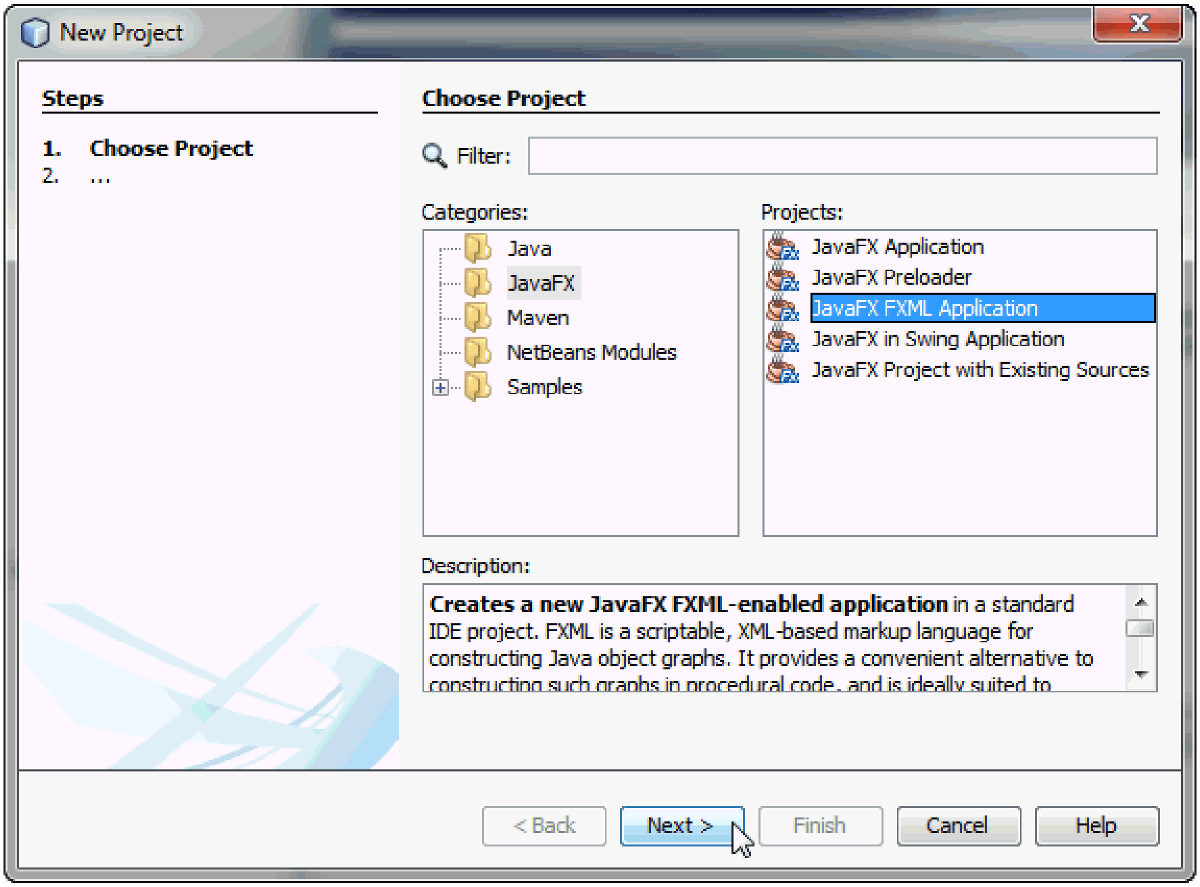
Go to http://netbeans.org/downloads for the latest NetBeans IDE 8 download and installation information.

Creating a New JavaFX FXML Project

In NetBeans IDE, you use the New wizard to create a new JavaFX FXML Application, which is a JavaFX project that is based on an FXML layout. After the project is created, you can edit the FXML file using Scene Builder.

1. From the NetBeans IDE Main menu, select **File**, and then choose **New Project**.
2. In the New Project dialog box, choose the **JavaFX** category and **JavaFX FXML Application** project, as shown in Figure 2–1. Click **Next**.

Figure 2–1 Create New JavaFX FXML Application Project



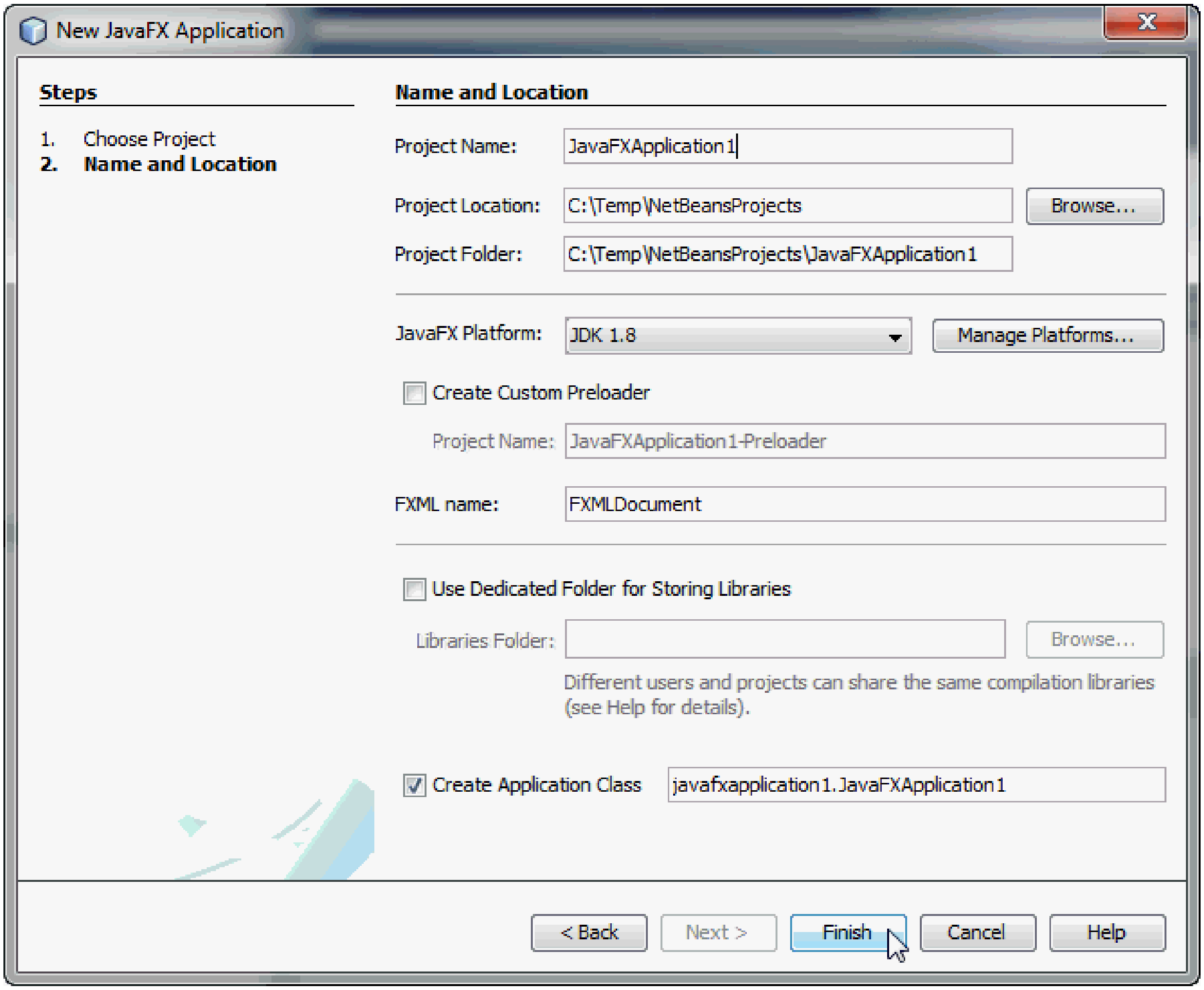
Categories: JavaFX

Projects: JavaFX FXML Application

Description: **Creates a new JavaFX FXML-enabled application** in a standard IDE project. FXML is a scriptable, XML-based markup language for constructing Java object graphs. It provides a convenient alternative to constructing such graphs in procedural code, and is ideally suited to

3. In the New JavaFX Application dialog box, enter the values you would like to use for the project name, project location, and FXML file name. Click **Finish** to complete the project creation. Figure 2–2 gives an example with sample values.

Figure 2–2 Set the Name and Location of the New JavaFX Application



New JavaFX Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: JavaFXApplication1

Project Location: C:\Temp\NetBeansProjects

Project Folder: C:\Temp\NetBeansProjects\JavaFXApplication1

JavaFX Platform: JDK 1.8

□ Create Custom Preloader

Project Name: JavaFXApplication1-Preloader

FXML name: FXMLDocument

□ Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☑ Create Application Class javafxapplication1.JavaFXApplication1

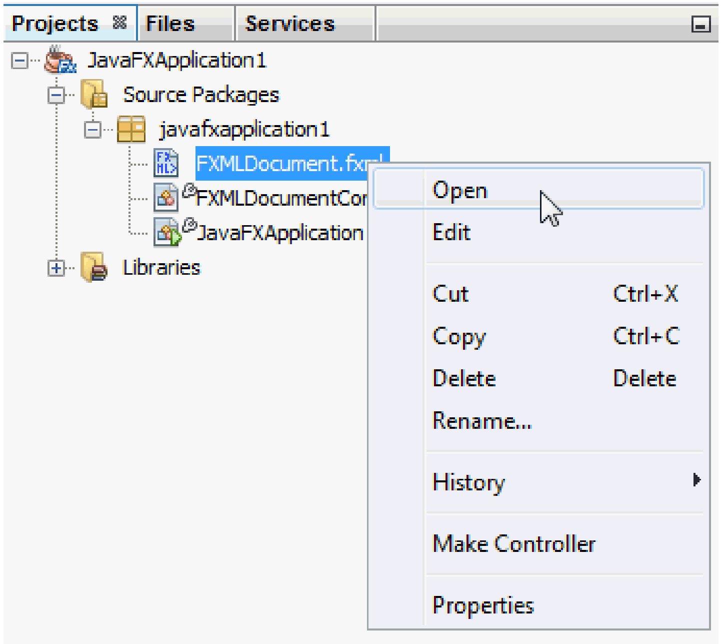
The IDE's Projects window is updated with the new JavaFX application you just created.

Editing an FXML File Using Scene Builder

You can modify your FXML file using the NetBeans IDE editor or by opening it using Scene Builder. If you installed Scene Builder in a location other than the default, you need to first ensure that NetBeans IDE is configured with that Scene Builder installation by following Configuring the Scene Builder Installation Location.

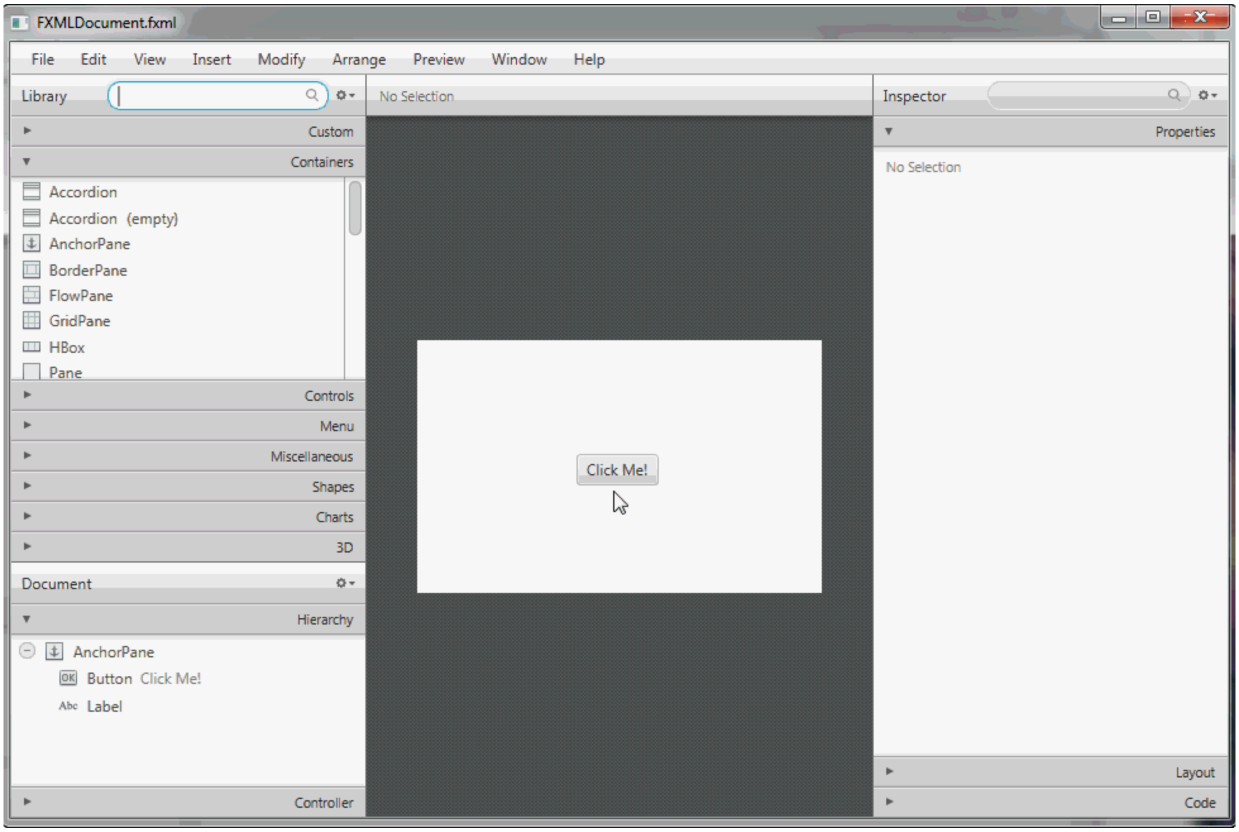
As shown in Figure 2–3, you can either choose **Open** to edit the FXML file with the Scene Builder tool or choose **Edit** to edit the FXML file with the Netbeans FXML editor.

Figure 2–3   Open FXML File from NetBeans IDE



When you right-click the node for the FXML file and choose **Open**, NetBeans IDE invokes the latest installed Scene Builder on your system, and the Scene Builder window appears, as shown in Figure 2–4. Changes you make to your FXML file using Scene Builder are reflected in the file.

Figure 2–4 Open FXML File in Scene Builder



Configuring the Scene Builder Installation Location

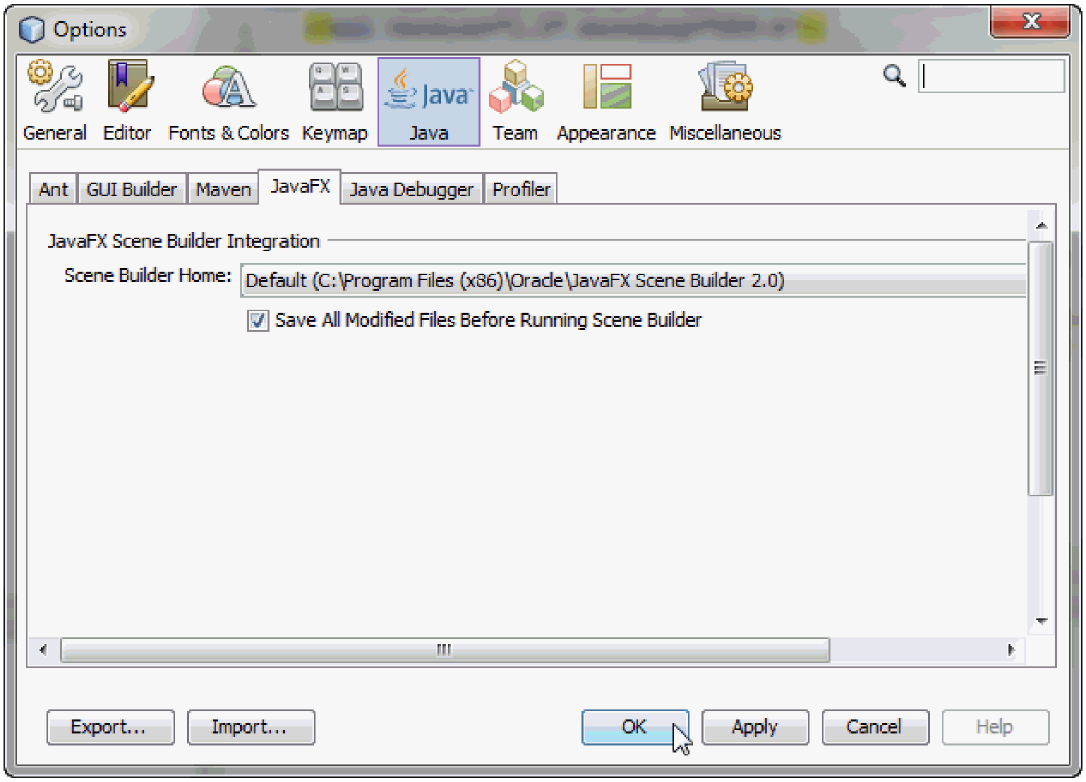
If you did not install Scene Builder in the default installation location, use the following steps to let NetBeans IDE be aware of its location:

1. From the IDE's **Main** menu, select **Tools** and choose **Options**.
   On the Mac OS platform, select **NetBeans** and choose **Preferences** from the **Main** menu.

2. In the Options window, click **Java** and then the **JavaFX** tab.

3. Specify the location of the Scene Builder installation folder and then click **OK**.
   Figure 2–5 shows the JavaFX tab with the path to the Scene Builder installation.

Figure 2–5 Set the Scene Builder Home in NetBeans IDE



JavaFX Scene Builder Integration

Scene Builder Home: Default (C:\Program Files (x86)\Oracle\JavaFX Scene Builder 2.0)

☑ Save All Modified Files Before Running Scene Builder

Synchronizing With the Controller Source Code

The NetBeans IDE's Make Controller feature allows you to synchronize the modifications you make in the FXML file that is currently opened in Scene Builder and the controller source code opened in NetBeans IDE. To illustrate this feature, do the following:

1. In the Scene Builder window, drag another Button control from the Library panel to the Content panel.
2. In the Code panel on the right side of the window, type `button2` for the new button's **`fx:id`** field and a new method name, `handleButtonAction2`, for the **`onAction`** method.
3. Select **File** in the main menu and then **Save**.
4. In NetBeans IDE, right click the node for the `FXMLDocument.fxml` file that you just edited, if it is not already opened, and select **Edit** from the contextual menu. Notice there is an error badge to indicate that the handleButtonAction2 is undefined.
5. From the IDE's main menu, select **Source** and then **Make Controller**.
   The `@FXML private Button button2` declaration and the declaration for the new method, `handleActionButton2`, are created in the controller source file, `FXMLDocumentController.java`. Use the **Make Controller** command if you delete an element from the Scene Builder's Content panel, or update an `fx:id` value or a method name in Scene Builder.

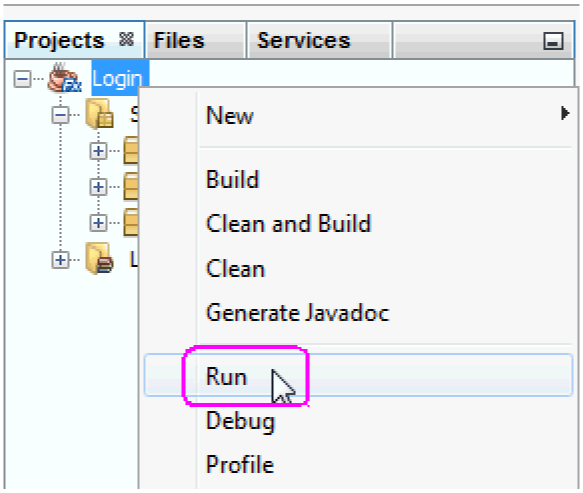
Where to Go From Here

Now that you are familiar with the integration between the Scene Builder tool and NetBeans IDE, look at `JavaFX Scene Builder User Guide` to learn more about the available Scene Builder features. Use the steps described in `Getting Started with JavaFX Scene Builder` to create a simple issue tracking application.

You can also try the sample applications provided with the Scene Builder release. These samples are Netbeans projects, which means they can be compiled and ran directly after being opened in the NetBeans IDE.

1. Download the Scene Builder samples bundle from the Additional Resources section of the Java SE Downloads page at http://www.oracle.com/technetwork/java/javase/downloads/index.html.
2. Extract the `javafx_scenebuilder_samples-2_0.zip` file.
3. From the NetBeans IDE, open on of the sample NetBeans projects you just extracted.
4. Right-click the project's node in the IDE's Project window and choose **Run**, as shown in Figure 2–6.

Figure 2–6 Run Scene Builder Sample Application



3

Using Scene Builder with Login Eclipse IDE

This chapter describes how to download and install the e(fx)clipse tool, which enables you to create a new JavaFX FXML project using Eclipse IDE, start Scene Builder from within the IDE, and run Scene Builder sample applications.

The JavaFX integration with the Eclipse IDE is provided with the e(fx)clipse tool. It is assumed that you have already installed Scene Builder before continuing with the rest of this section. Go to the JavaFX Scene Builder Installation Guide for installation information.

Downloading and Installing e(fx)clipse

Go to `http://www.eclipse.org/efxclipse/install.html` for information about the available options to download and install the e(fx)clipse tool to use with your JavaFX Scene Builder installation.

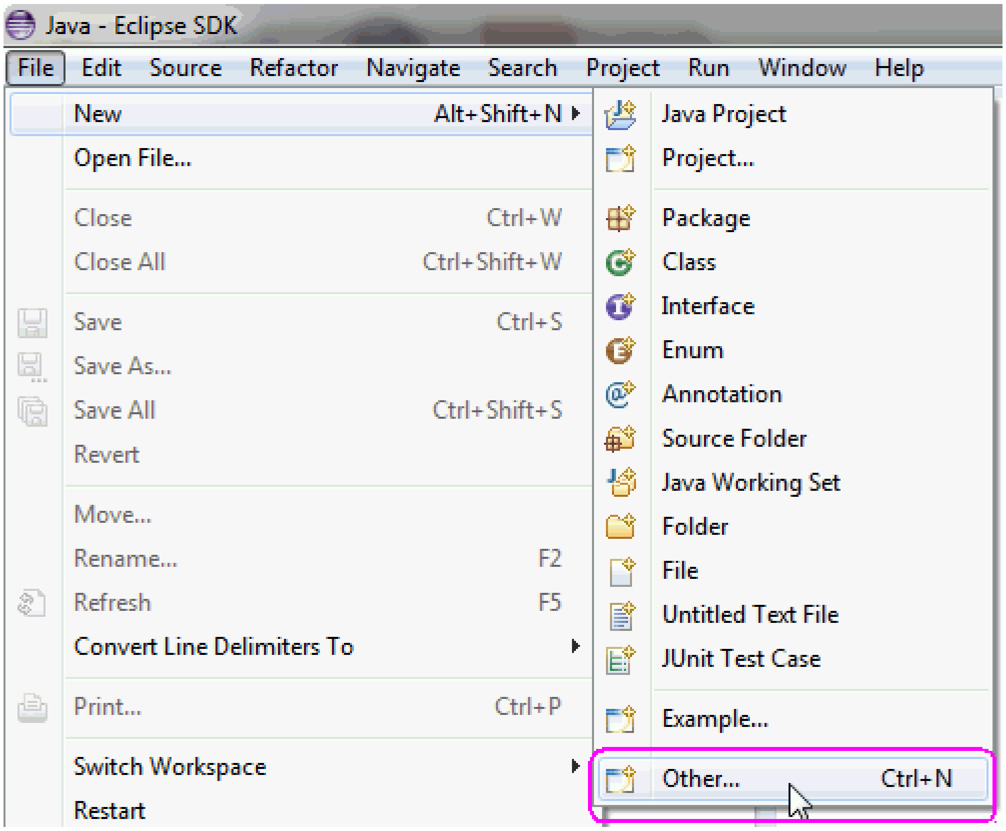
> **Note:** The information in this chapter is based on the pre-configured distribution of Eclipse 4.3.1 SDK with e(fx)clipse 0.9.0.201401250805

Creating a New JavaFX FXML Project

To create a new JavaFX FXML project using Eclipse, you must first create a new JavaFX project and then add a new FXML document to that JavaFX project.

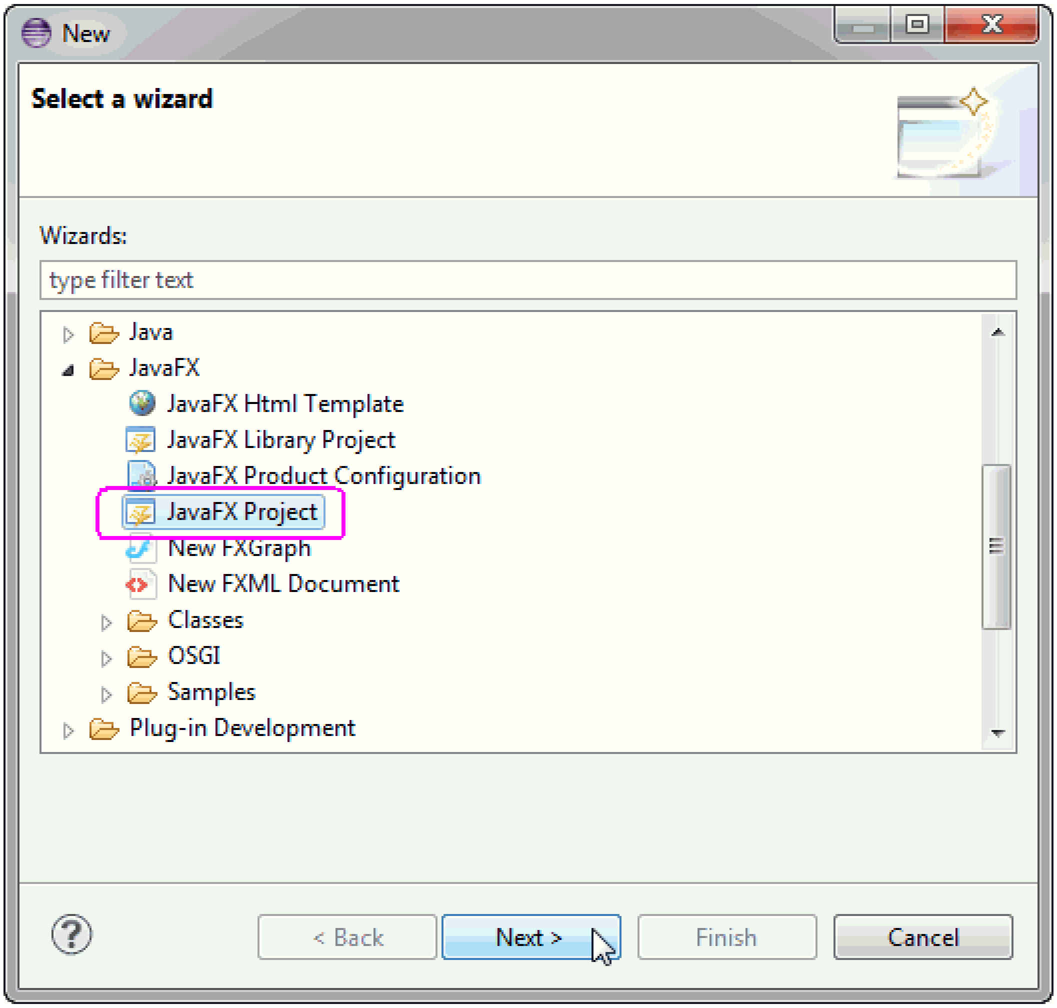
1. From the Eclipse IDE Main menu, select **File**, choose **New,** and then **Other**, as shown in Figure 3–1.

Figure 3–1 Create a New File



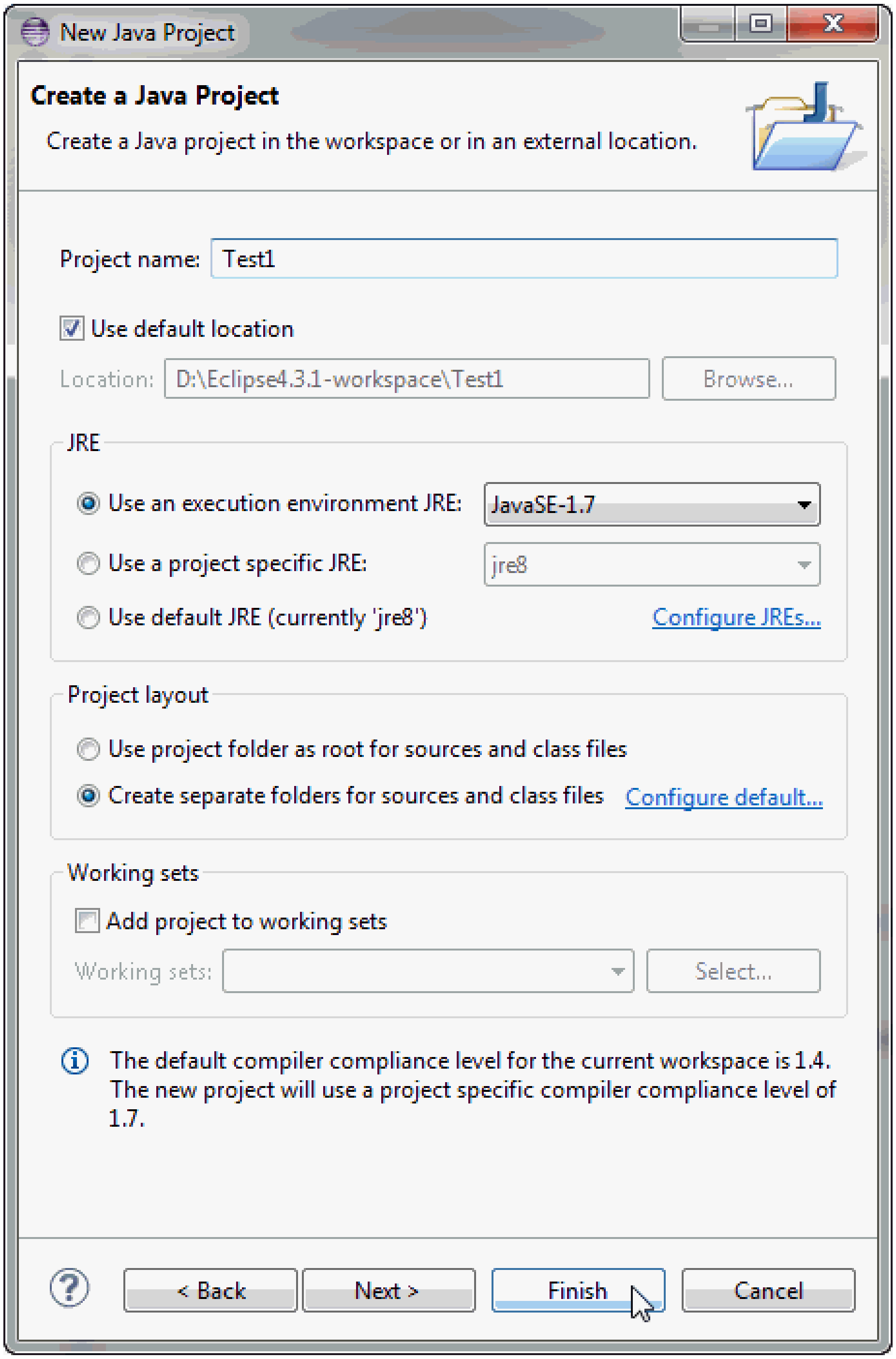
2. From the New dialog box, expand the JavaFX folder, and select the **JavaFX Project** wizard, as shown in Figure 3–2, and click **Next**.

Figure 3–2 Choose the JavaFX Project Wizard



3. Enter the project name, specify the JRE to use, and select the Project layout option. Click **Finish** to complete the project creation, as shown in Figure 3–3.

Figure 3–3 Create a New Java Project



New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: Test1

☑ Use default location

Location: D:\Eclipse4.3.1-workspace\Test1

JRE

● Use an execution environment JRE: JavaSE-1.7

○ Use a project specific JRE: jre8

○ Use default JRE (currently 'jre8')

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files

Working sets

□ Add project to working sets

Working sets:

The default compiler compliance level for the current workspace is 1.4. The new project will use a project specific compiler compliance level of 1.7.

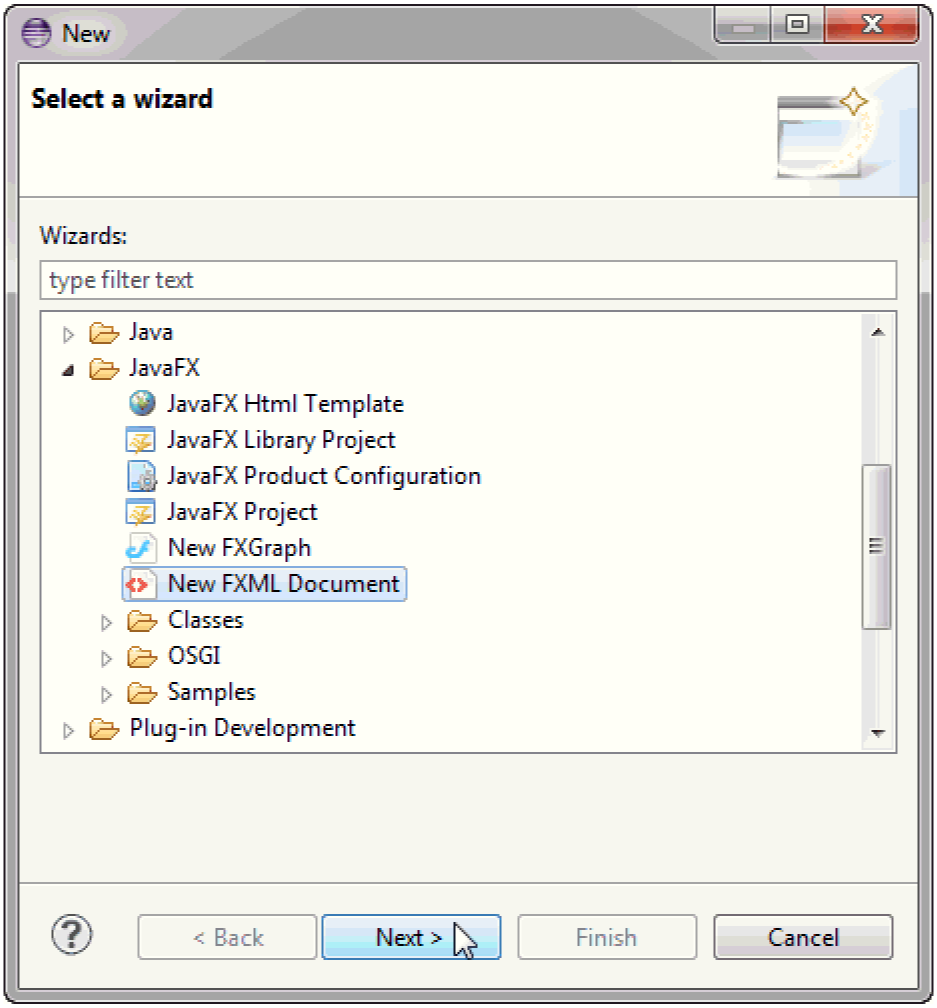
4. Now create a new FXML document.

   a. Right-click the **Test1** folder in the Package Explorer.

   b. Select **New** and then **Other**.

   c. Expand the JavaFX folder in the New dialog box and then choose **New FXML Document**, as shown in Figure 3–4. Click **Next**

Figure 3–4 Create a New FXML Document



5. In the FXML File dialog box, complete the creation of the new FXML document using the following steps:

   a. If not automatically set to `Test1 /src`, specify the Source folder value by clicking **Browse**. Navigate to the `src` folder of the JavaFX project that you just created in the previous steps. Leave the Package field as is.

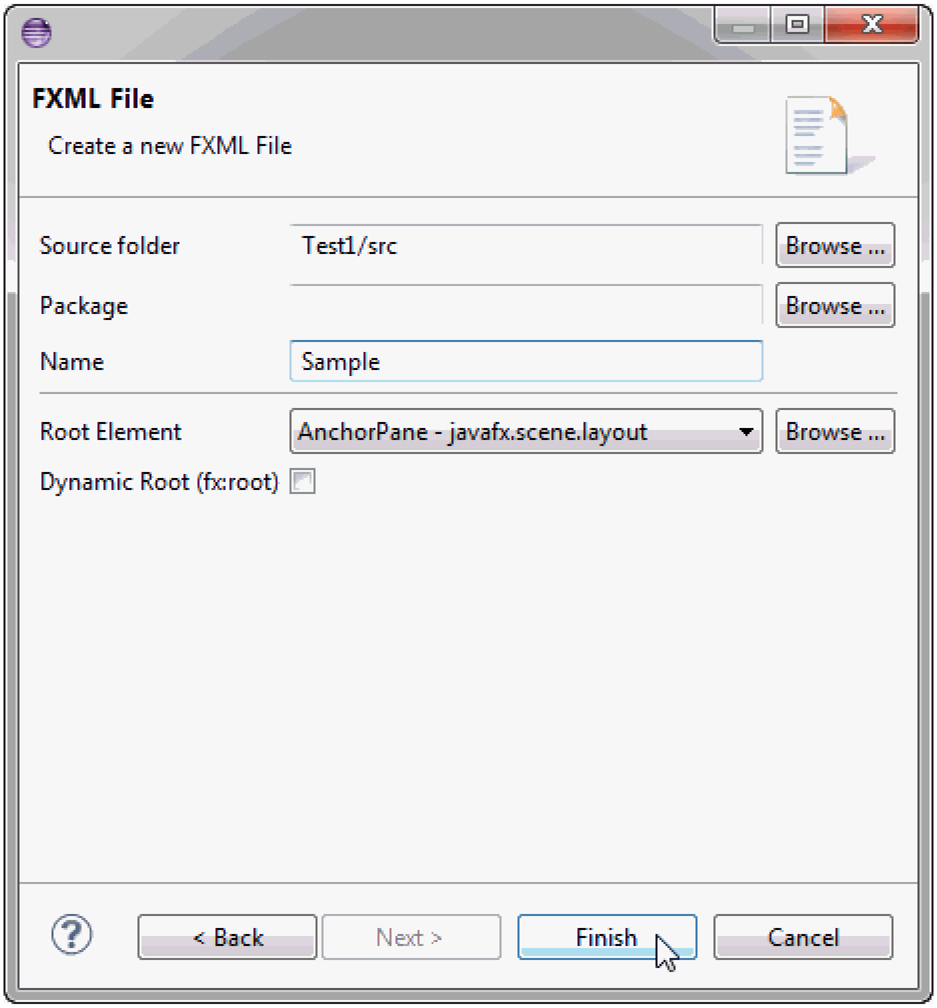
   b. In the Name field, enter `Sample` for the FXML file name, as shown in Figure 3–5.

   c. If not already set, specify the FXML document's root element by clicking **Browse** and selecting an item from the Find Preloader dialog box. The default value shown is AnchorPane.

   d. Leave Dynamic Root unselected. It is for advanced users who want to manage custom types.

   e. Click **Finish** to close the dialog box.
   You can now complete the creation of your JavaFX FXML layout.

Figure 3–5 Finish Creating a New FXML File



FXML File

Create a new FXML File

Source folder: Test1/src

Package:

Name: Sample

Root Element: AnchorPane - javafx.scene.layout

Dynamic Root (fx:root) □

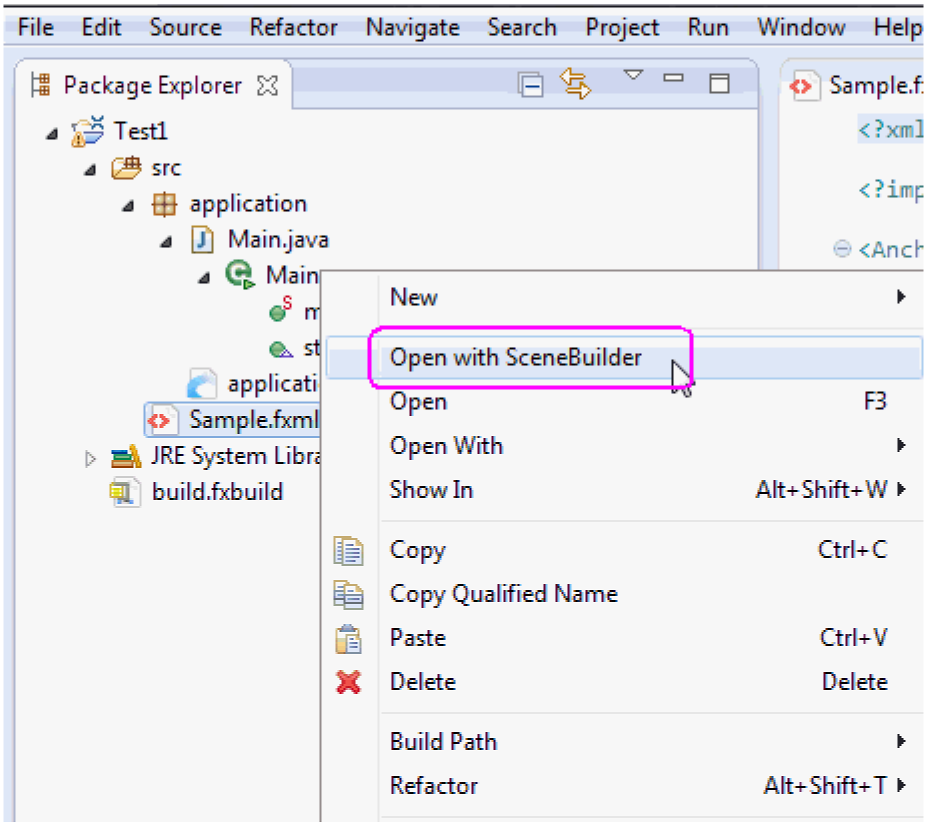
Editing an FXML File Using Scene Builder

You can edit an FXML file using the Eclipse FXML editor or by opening the file using the JavaFX Scene Builder tool:

1. In the IDE's Package Explorer tab, expand the **Test1** and **src** folders.
2. Right-click the node for the `Sample.fxml` file and select **Open with Scene Builder**, as shown in Figure 3–6.

   A separate window for Scene Builder is opened and displays the `Sample.fxml` layout in the Content panel area. You may have to increase the size of the default AnchorPane that appears in the Content panel. You may start to add controls.

Figure 3–6 Open Sample.fxml File in Scene Builder



3. Use JavaFX Scene Builder User Guide to learn more about the available Scene Builder features and Getting Started with JavaFX Scene Builder to create a simple issue tracking application.

Where to Go From Here

Now that you are familiar with the integration between the Scene Builder tool and Eclipse IDE, look at JavaFX Scene Builder User Guide to learn more about the available Scene Builder features. Follow the steps described in Getting Started with JavaFX Scene Builder to create a simple issue tracking application.

You can also try the sample applications provided with the Scene Builder release. The samples are Netbeans projects, so use the following information to learn how to use a sample using Eclipse IDE.

Using Scene Builder Samples From Eclipse IDE

Use the following information to use the Scene Builder samples. The Login application is used for illustratative purpose.

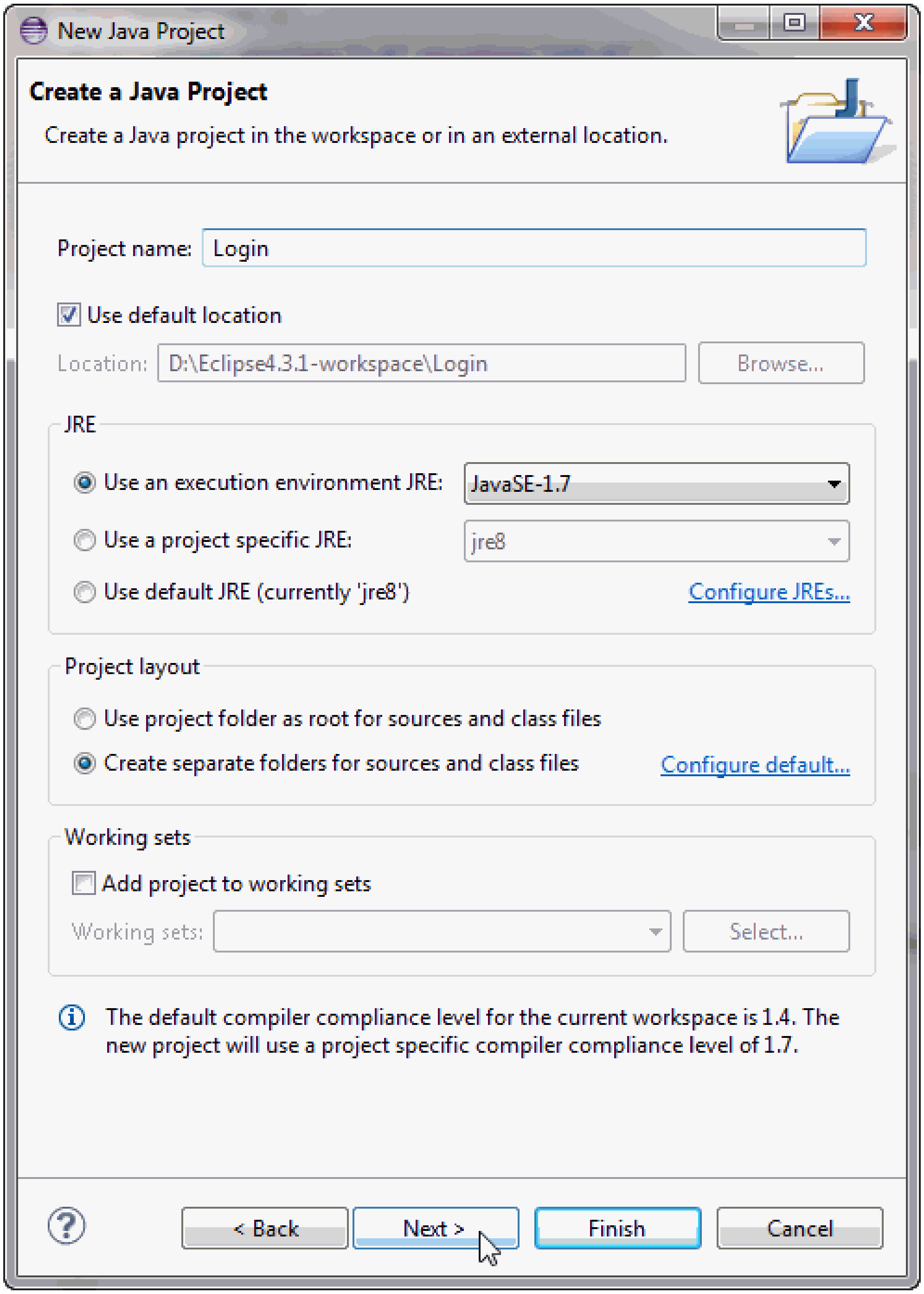
1. Ensure that you have downloaded JavaFX Scene Builder Samples bundle file from the Additional Resources section of the Java SE Download page at `http://www.oracle.com/technetwork/java/javase/downloads/index.html` and have extracted its contents.

2. Create a new JavaFX Project by selecting **File**, and then choosing **New** and then **Other**.

3. In the New wizard, expand the JavaFX folder and choose **JavaFX Project**. Click **Next**.

4. In the New Java Project dialog box, enter Login for the **Project Name**, as shown in Figure 3–7, and click **Next**.

Figure 3–7 Create a JavaFX Project for the Login Sample



New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: Login

☑ Use default location

Location: D:\Eclipse4.3.1-workspace\Login [Browse...]

JRE

● Use an execution environment JRE: JavaSE-1.7

○ Use a project specific JRE: jre8

○ Use default JRE (currently 'jre8') Configure JREs...

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files Configure default...

Working sets

□ Add project to working sets

Working sets: [Select...]

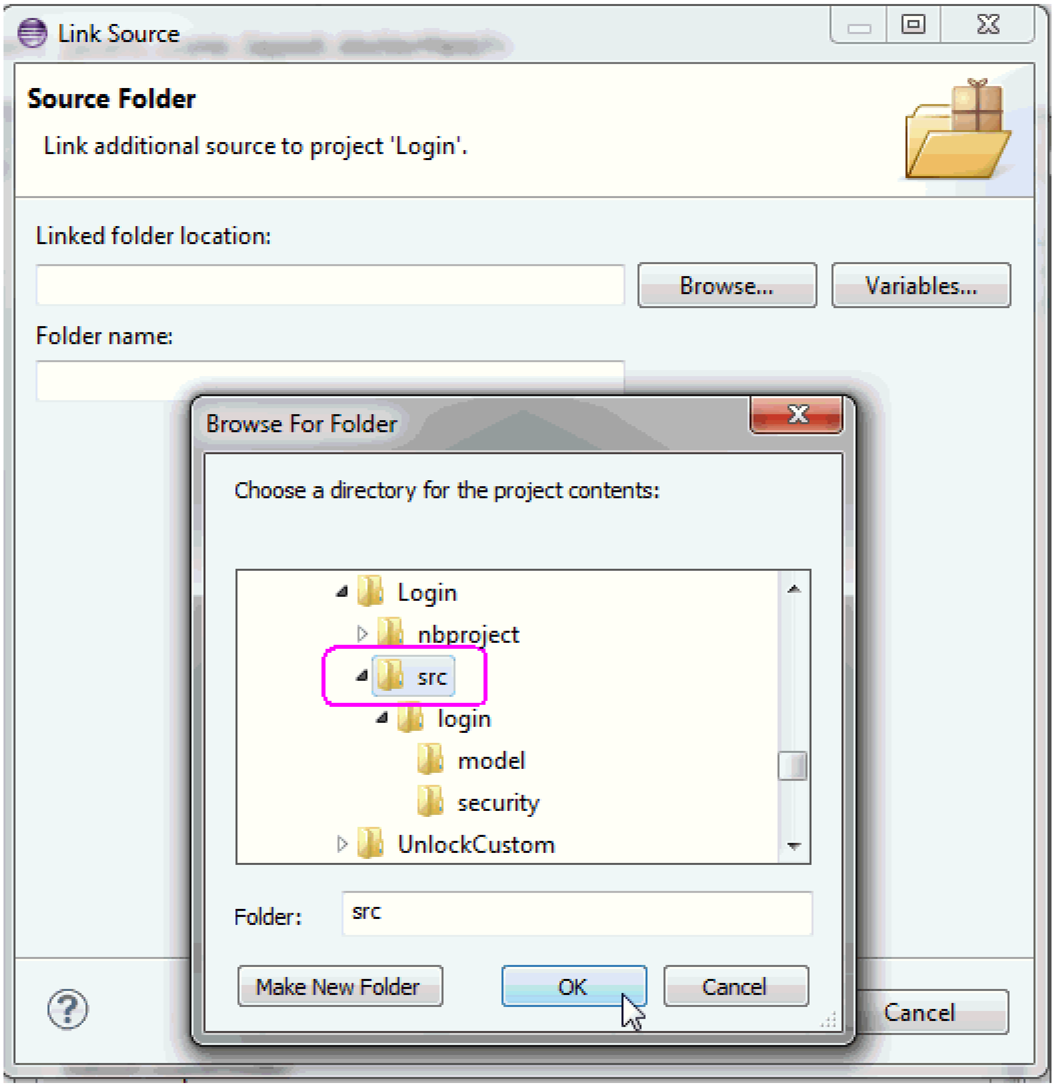
The default compiler compliance level for the current workspace is 1.4. The new project will use a project specific compiler compliance level of 1.7.

5. In the Details section of the Java Settings dialog box, click **Link additional source** option. The Link Source dialog box is displayed.

6. In the Link Source dialog box, click **Browse** and navigate to the location of the src folder for the Scene Builder Login sample application and click **OK**, as shown in Figure 3–8.

> **Note:** Ignore the error "The folder is already a source folder." because the Folder name field is automatically filled in

Figure 3–8 Open the Login Sample's src Folder



7. Change the default value in the Folder name text field to `src_samples`, or any folder name of your choice, and click **Finish**.

8. Back in the New Java Project wizard, click **Finish**.

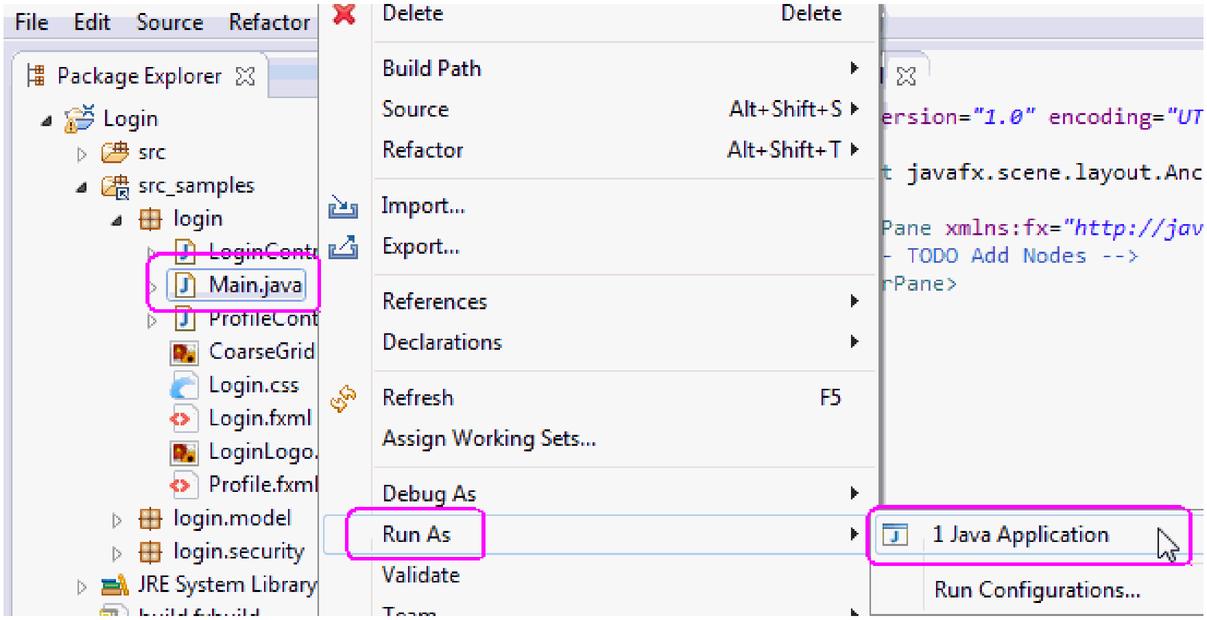
9. Run the Login sample, as shown in Figure 3–9:

   a. In the Project Explorer window, expand the `src_samples` and `login` folders for the Login project.

   b. Right-click the node for the `Main.java` source file.

c. Choose **Run As** and then **Java Application**.

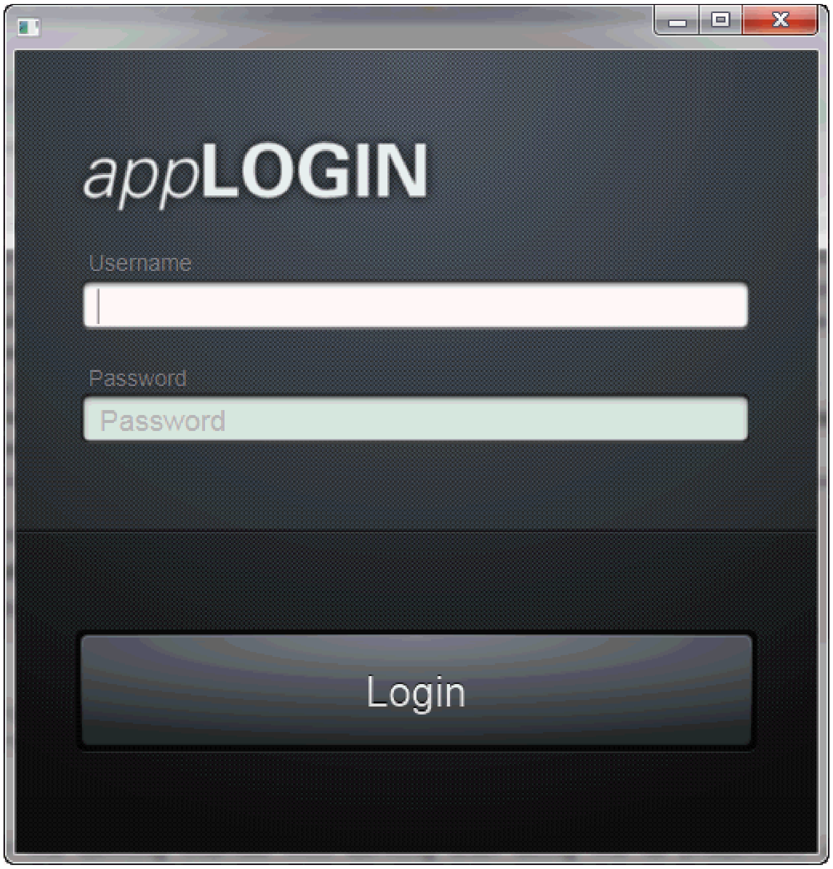
Figure 3–9 Run Scene Builder Login Sample



```
ersion="1.0" encoding="UT

t javafx.scene.layout.Anc

Pane xmlns:fx="http://jav
- TODO Add Nodes -->
rPane>
```

The Login application is displayed, as shown in Figure 3–10.

Figure 3–10 Login Sample Running



4

Using Scene Builder with IntelliJ IDEA

This chapter gives information on how to download and use the JavaFX 2 plug-in that provides the JavaFX integration with the IntelliJ IDEA 13 Community Edition.

The JavaFX 2 plug-in enables you to create a new JavaFX Application project, start Scene Builder from within the IDE, and run the Scene Builder Login sample application. It is assumed that you have already installed Scene Builder before continuing with the rest of this chapter. Go to the JavaFX Scene Builder Installation Guide for installation information.

Downloading and Installing IntelliJ IDEA

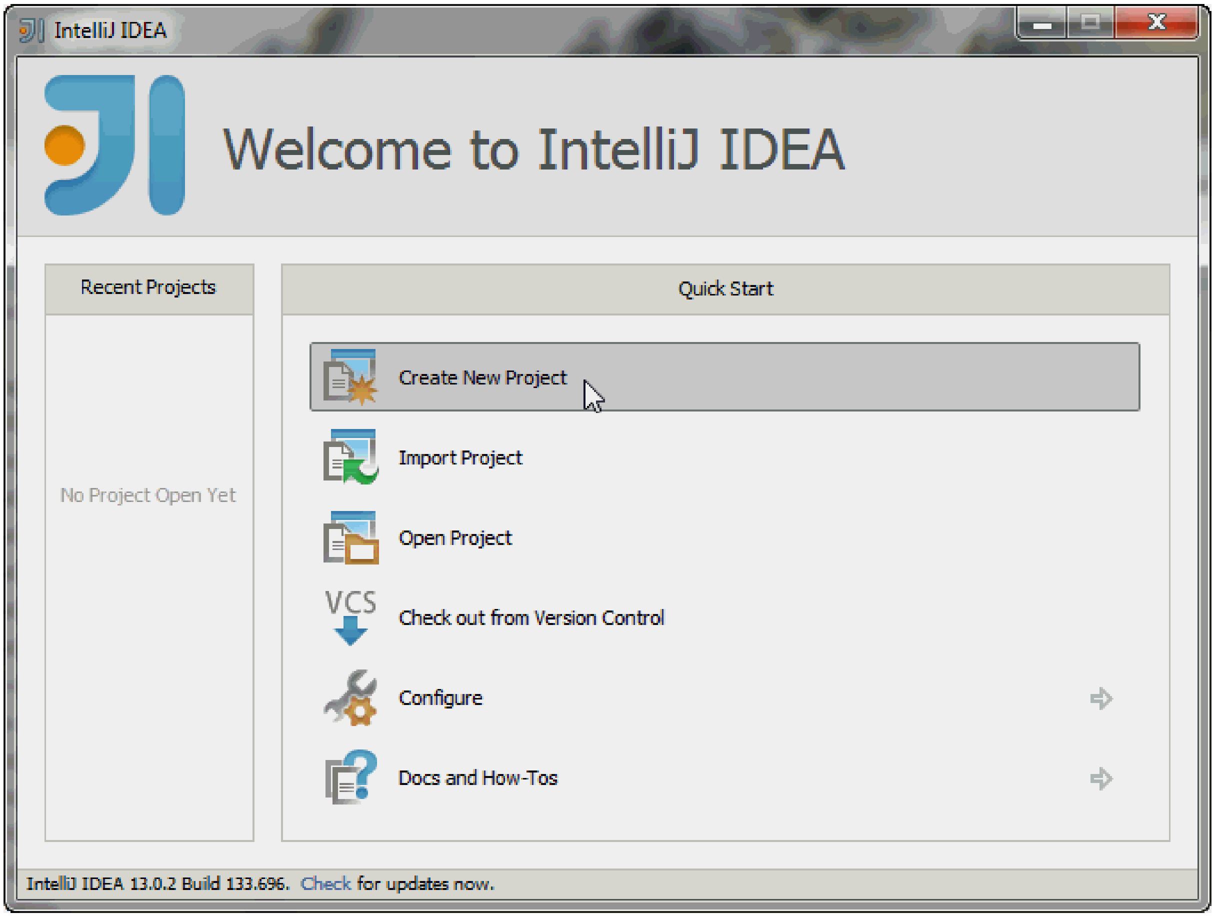
Use the information from `http://www.jetbrains.com/idea/download/` to download and install the latest IntelliJ IDEA 13 version that includes the JavaFX plug-in.

Creating a New JavaFX Application

The JavaFX plug-in for IntelliJ IDEA 13 enables you to create a new JavaFX application template.

1. From the IntelliJ IDEA welcome window, click the **Create New Project**, as shown in Figure 4–1.

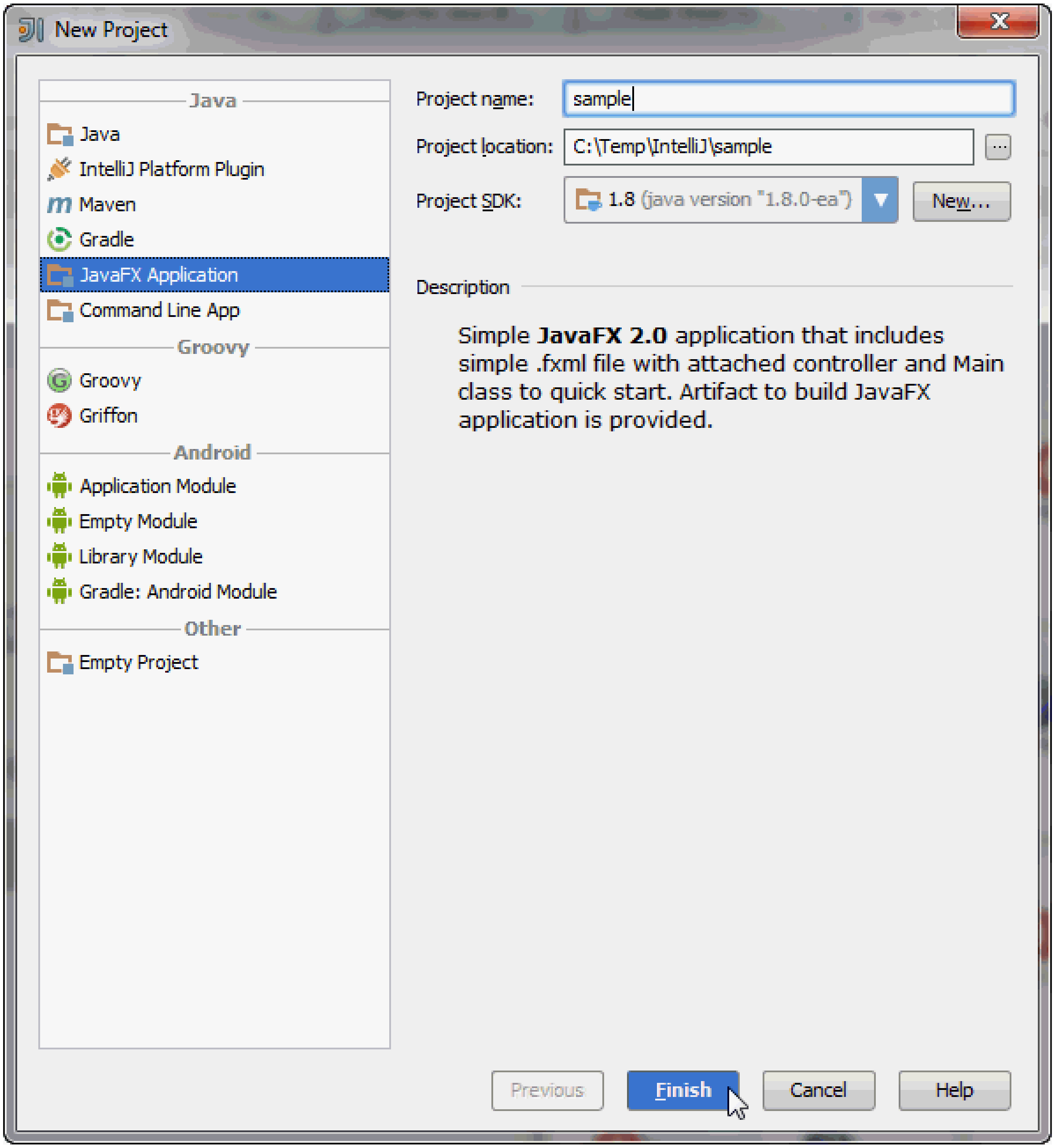
Figure 4–1 Select Create New Project in the Welcome Window



2. In the New Project dialog box, select **JavaFX Application**, specify the project information, as shown in Figure 4–2, and click **Finish**.

   The node for a new JavaFX application template is added to the Project tab in the main window.

Figure 4–2 Specify the Project Information for the New JavaFX Application Project



Project name: sample

Project location: C:\Temp\IntelliJ\sample

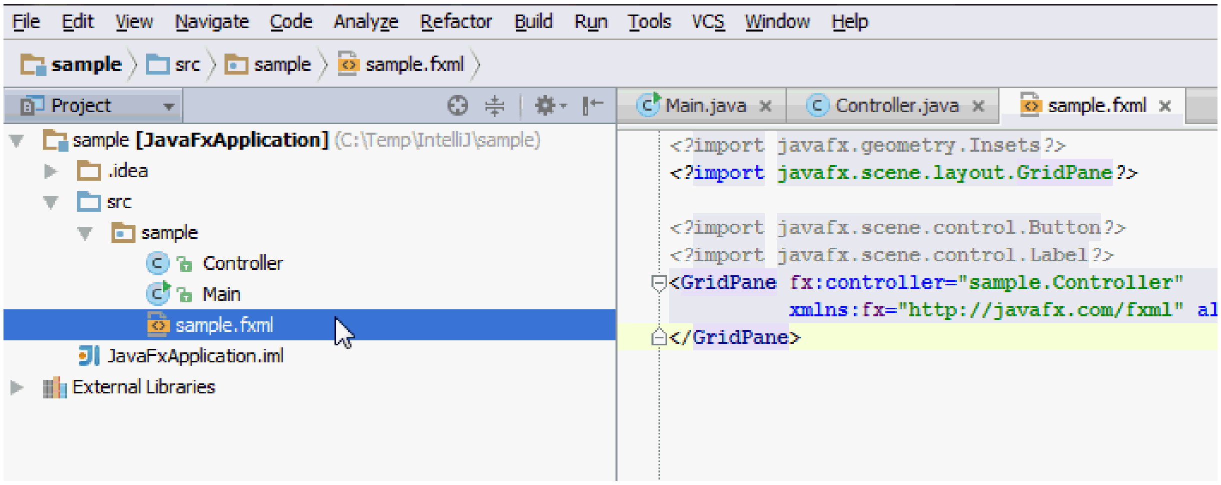
Project SDK: 1.8 (java version "1.8.0-ea")

Description

Simple **JavaFX 2.0** application that includes simple .fxml file with attached controller and Main class to quick start. Artifact to build JavaFX application is provided.

3. Expand the nodes for the project, as shown in Figure 4–3.

Figure 4–3 Expand the Nodes in the Project Window


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
sample 〉 src 〉 sample 〉 sample.fxml 〉
Project
Main.java ×
Controller.java ×
sample.fxml ×
sample [JavaFxApplication] (C:\Temp\IntelliJ\sample)
.idea
src
sample
Controller
Main
sample.fxml
JavaFxApplication.iml
External Libraries

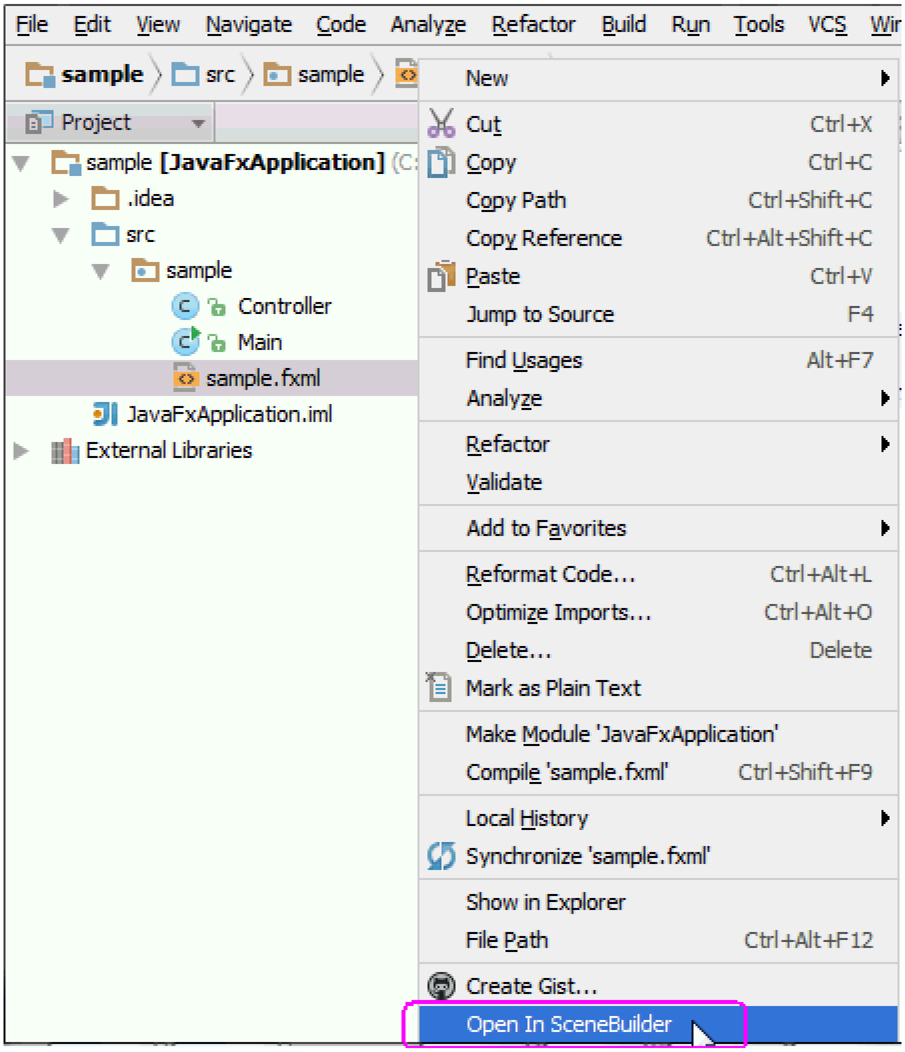
```
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<GridPane fx:controller="sample.Controller"
          xmlns:fx="http://javafx.com/fxml" al
</GridPane>
```

Starting Scene Builder from IntelliJ IDE

Included in a newly created JavaFX Application template is the sample.fxml, which you can open and edit using Scene Builder.

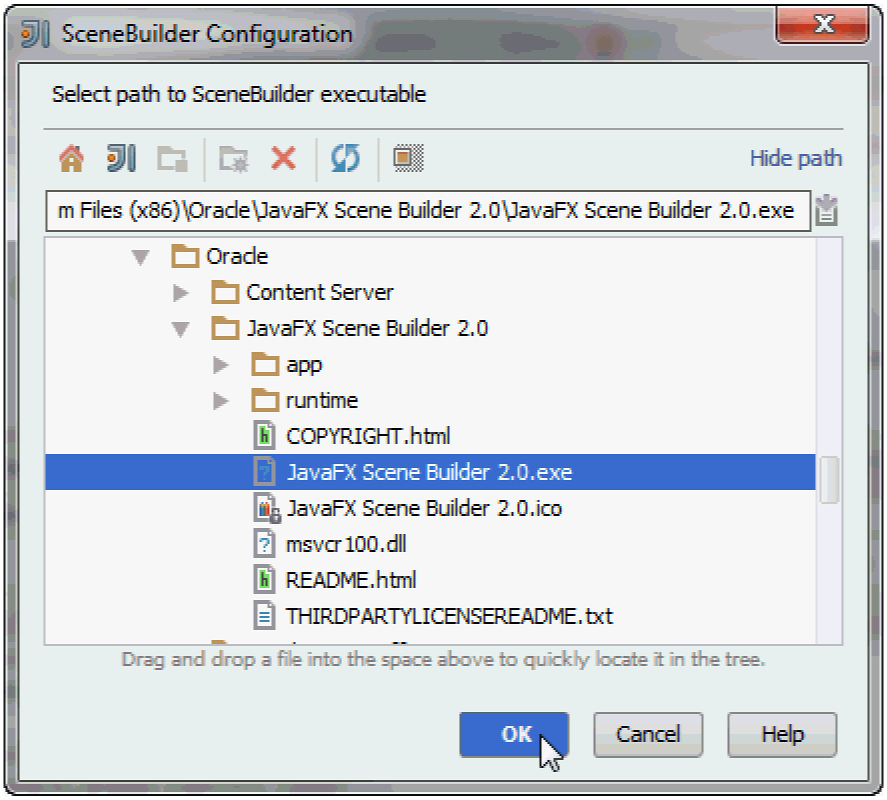
1. In the Projects tab, right-click the `sample.fxml` file and select **Open In Scene Builder**, as shown in Figure 4–4.

Figure 4–4 Open JavaFX Scene Builder to sample.fxml File



2. In the Scene Builder Configuration dialog window, specify the path to your Scene Builder installation the first time you try to open Scene Builder from within IntelliJ IDEA and click **OK**, as shown in Figure 4–5.

Figure 4–5 Select Path to the Scene Builder Installation Folder



SceneBuilder Configuration

Select path to SceneBuilder executable

m Files (x86)\Oracle\JavaFX Scene Builder 2.0\JavaFX Scene Builder 2.0.exe

```
▼ Oracle
  ▶ Content Server
  ▼ JavaFX Scene Builder 2.0
    ▶ app
    ▶ runtime
      COPYRIGHT.html
      JavaFX Scene Builder 2.0.exe
      JavaFX Scene Builder 2.0.ico
      msvcr100.dll
      README.html
      THIRDPARTYLICENSEREADME.txt
```

Drag and drop a file into the space above to quickly locate it in the tree.

3. Use the JavaFX Scene Builder User Guide document to learn more about the available Scene Builder features and the Getting Started with JavaFX Scene Builder tutorial to create a simple issue tracking application.

Where to Go From Here

Now that you are familiar with the integration between the Scene Builder tool and IntelliJ IDEA, look at JavaFX Scene Builder User Guide to learn more about the available Scene Builder features. Also, use the Getting Started with JavaFX Scene Builder to create a simple issue tracking application.

You can also try running the sample applications provided with the Scene Builder release. These samples are Netbeans projects, so use the information in the following section to learn how to use one of the samples using the IntelliJ IDEA.

Running Scene Builder Samples from IntelliJ IDEA

Use the following steps to run the Login sample that is included in the JavaFX Scene Builder samples bundle:

1. Ensure that you have downloaded JavaFX Scene Builder Samples bundle file from the Additional Resources section of the Java SE Download page at `http://www.oracle.com/technetwork/java/javase/downloads/index.html` and extracted its contents.
2. From the IntelliJ IDEA main window, click **File** and choose **Import Project**.

3. Navigate to the location where you extracted the JavaFX Scene Builder samples and expand the **Login** sample folder. Select the **src** folder and click **OK**.

4. In the Import Project wizard, keep the "Create project from existing sources" option selected and click **Next**.

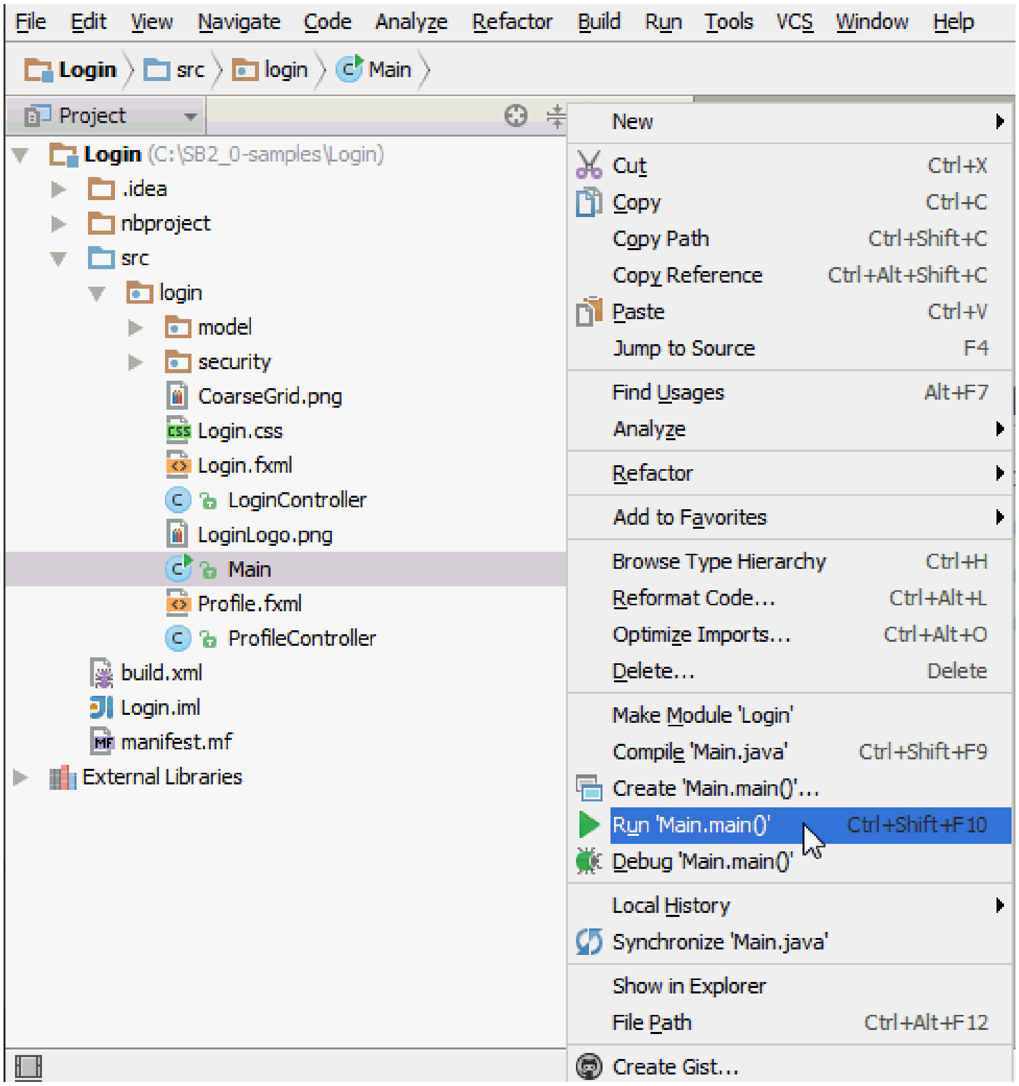
5. Type `Login` in the Project Name field, if it is not already filled in, and click **Next**.

6. Leave the rest of the default choices and values specified in the Import Project wizard and continue to click **Next** until you see the Finish button. Click **Finish**.

7. Choose to open the new project in the current window or open a new window. The new Login project is opened in the Project tab.

8. In the Project tab, right-click the node for the Login project's Main class and choose **Run Main.main()** as shown in Figure 4–6.

Figure 4–6 Run the Main Class File for Login Sample Application



The window for the Login sample application appears on top of the IDEA window, similar to what is shown in Figure 3–10.