The Practice and Philosophy of Object-Oriented Programming in Java

How to Add JavaFX to an IntelliJ Project

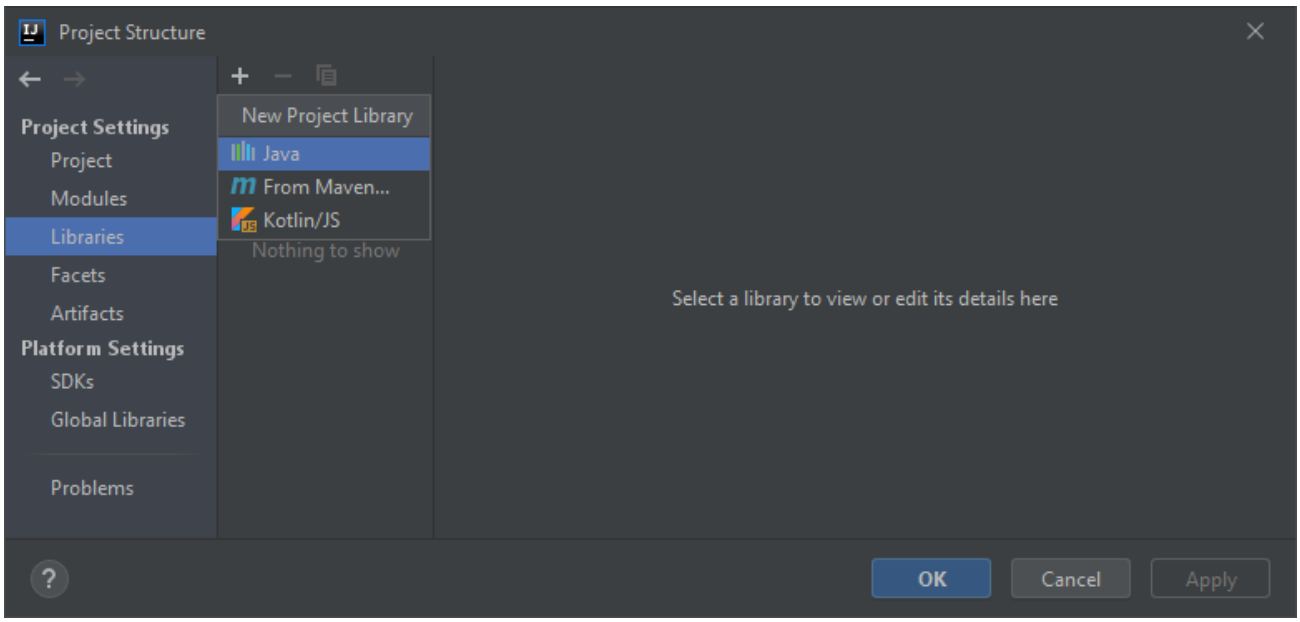
This document assumes that you have already installed and set up Java and IntelliJ IDEA (if not, start here). If you are using Java 8, JavaFX is already bundled with your JDK so you do not need to take any further steps. With the advent of Java 11 in 2018, however, JavaFX was removed from the JDK so that it could evolve at its own pace as an independent open-source project guided by Oracle and others in the OpenJFX community.

Step 1: Download JavaFX

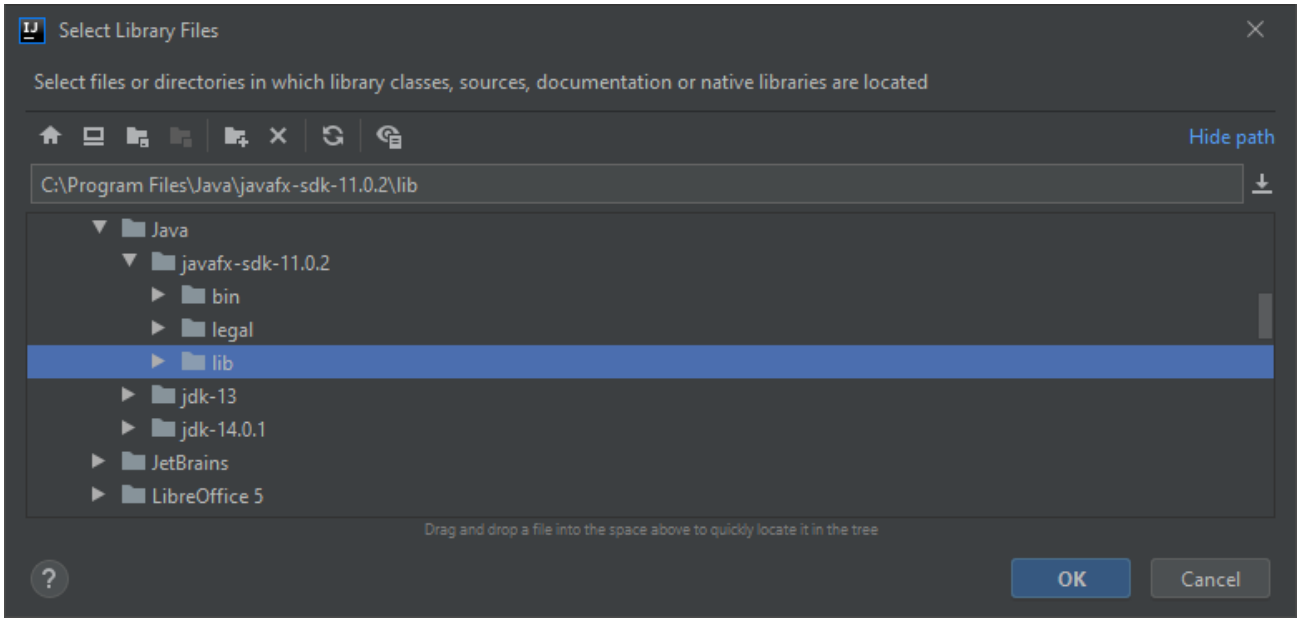
1. Download JavaFX. This is a zipped folder named **`javafx-sdk`** (with version number appended). Unzip the folder and move it to a permanent location on your hard drive. (Suggestion: move it to the folder that contains your JDK. On a Windows machine, this is **`C:\Program Files\Java`** unless you specified a different location when installing the JDK.)

Step 2: Adding the JavaFX library to an IntelliJ project

If you want to create a JavaFX application in an IntelliJ project, you will need to add the JavafX library. Click on the **`File`** menu and select **`Project Structure`**. In the dialog that appears, select the **`Libraries`** tab and click the **`+`** icon to add a new Java library:



Find your **`javafx-sdk`** folder and select the **`lib`** subfolder:



Click the OK button to complete the process.

Now right-click on a package and select **`new`**. You will see an option to create a new JavaFX application. Select this option. A code template will appear in the editor. Complete the **`start`** method with the instructions that you would like to execute. Now try to run the application. You will see that a **`ClassNotFound`** exception is thrown. The fourth and final step of the set-up process, described next, will fix this problem.

Step 3: Editing the Project's Run Configuration

From the **`run`** menu, select **`edit configurations`**. In the dialog that appears, you will see a field for VM options. You need to enter two options into this field with the following general form:

```
--module-path %PATH_TO_FX%
--add-modules=%FX_MODULES_NEEDED%
```

Instead of **%PATH_TO_FX%**, include the actual path to the JavaFX SDK **`lib`** directory. The path syntax will depend on your operating system. For the example illustrated in the preceding steps, the first VM option would be:

```
--module-path "C:\Program Files\Java\javafx-sdk-11.0.2\lib"
```

The quote marks are necessary if you are using Windows since the name of one of the directories in the path contains a space.

The JavaFX SDK is partitioned into seven modules. Include a comma-separated list of the ones your project will need in place of **%FX_MODULES_NEEDED%**. For example:

```
--add-modules=javafx.base,javafx.controls,javafx.graphics,javafx.media
```

Later you can add other modules if you need to, or remove some, by editing your run configuration again.

You can now run the application.

You will need to add the VM options for every JavaFX application in the current project. However, you can configure IntelliJ to automatically include the VM options for all future projects as follows. From the top menu bar, select **`File`** → **`New Projects Setup`** → **`Run Configurations Template`**. In the pop-up window that appears, select **`Application`** → **`Modify options`** → **`Add VM options`** and enter the options as described above.

Return to Companion Site home page