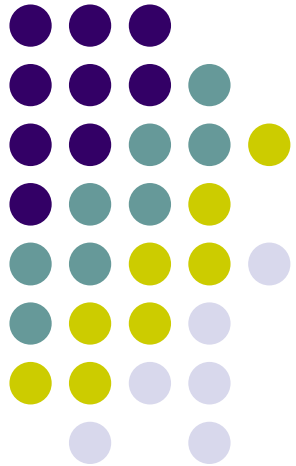
Setup JavaFX with JDK 11

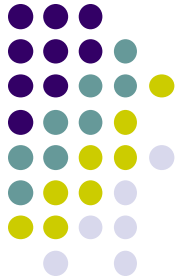
Downloads

JDK 11 Documentation

JavaFX Windows SDK SceneBuilder

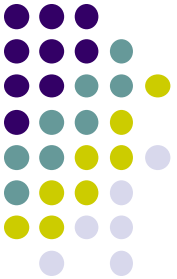


IntelliJ setup



Define the JDK in IntelliJ IDEA

- Open the Project Structure dialog (e.g. Ctrl+Shift+Alt+S).
- In the leftmost pane, under Platform **Settings**, click SDKs.
- Above the pane to the right, click + and select **JDK**.
- In the dialog that opens, select the installation directory of the **JDK** to be used and click OK

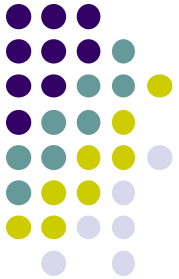


IntelliJ setup

Define the JDK in IntelliJ IDEA

- Open the Project Structure dialog (e.g. Ctrl+Shift+Alt+S).
- In the leftmost pane, under Platform **Settings**, click SDKs.
- Above the pane to the right, click + and select **JDK**.
- In the dialog that opens, select the installation directory of the **JDK** to be used and click OK

  (C:\Program Files\Java\jdk-11.0.1)

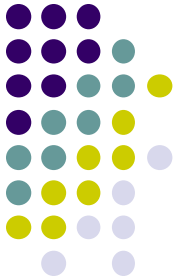


IntelliJ setup

Setup SceneBuilder

- Open the Settings dialog (e.g. Ctrl+Alt+S).
- In the leftmost pane, under Platform **Languages&Frameworks**, click JavaFX.
- On the right side locate and set the path to the SceneBuilder executable.

By default it is found in
C:\Users\<username>\AppData\Local\SceneBuilder\SceneBuilder.exe

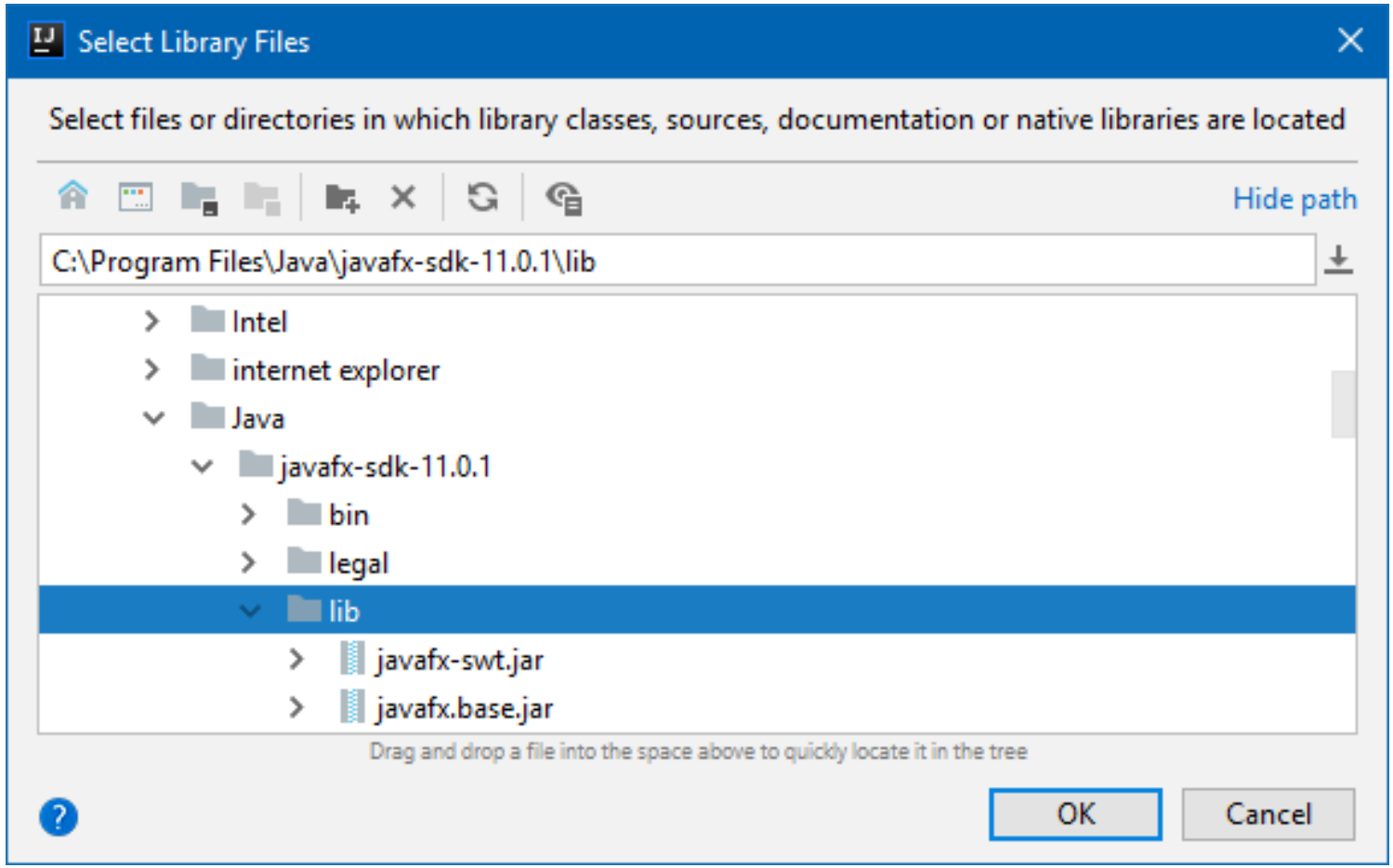


IntelliJ setup

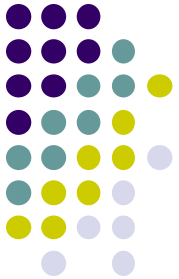
Setup JavaFX with JDK 11 as a **Global library**

- Open the Project Structure dialog (e.g. Ctrl+Shift+Alt+S).
- Select **Global Libraries**
- **Click + to add for Java the location of** the **lib** directory (Library-> Java) where you have unpacked JavaFX (for me, C:\Program Files\Java\javafx-sdk-11.0.1\lib).
-

IntelliJ setup



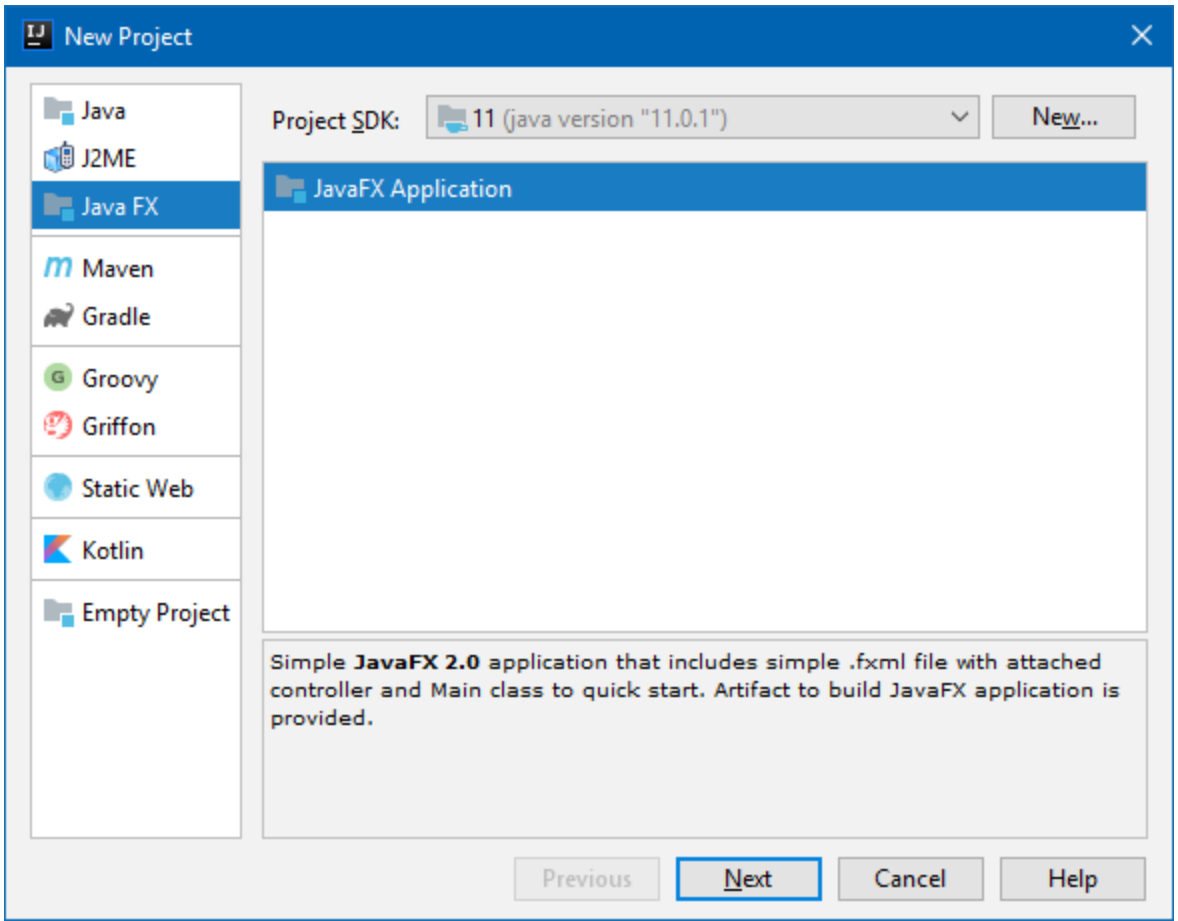
IntelliJ setup



Assign a descriptive name for the Global library, for example JavaFX

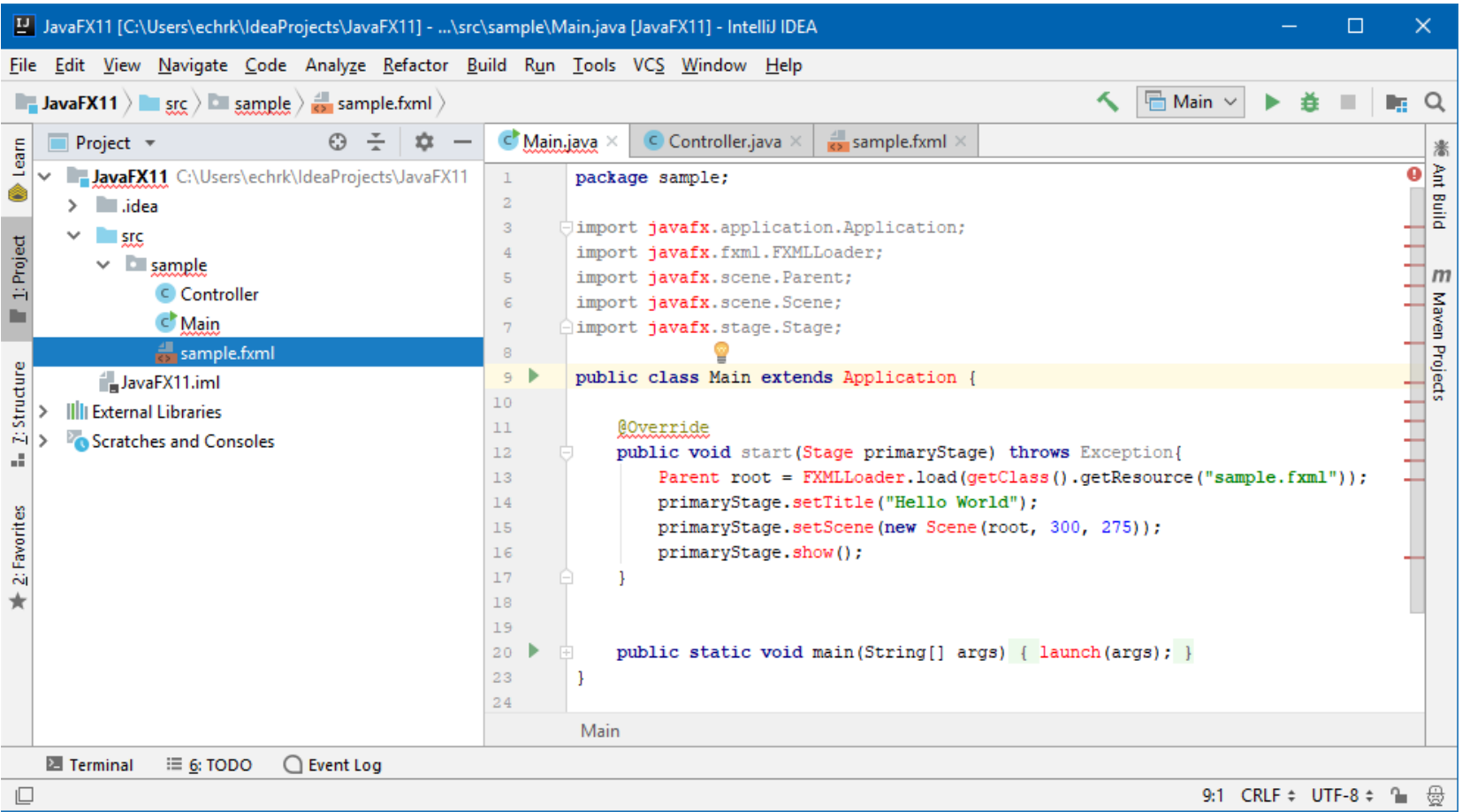
Non-Modular JavaFX project

Create a JavaFX project in IntelliJ in JDK 11.
Use JDK 11

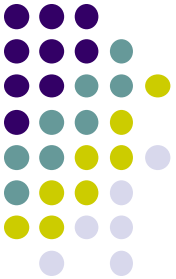


Non-Modular JavaFX project

Initially JavaFX 11 is not recognized



Non-Modular JavaFX project



Select **File->Project Structure->Project structure**

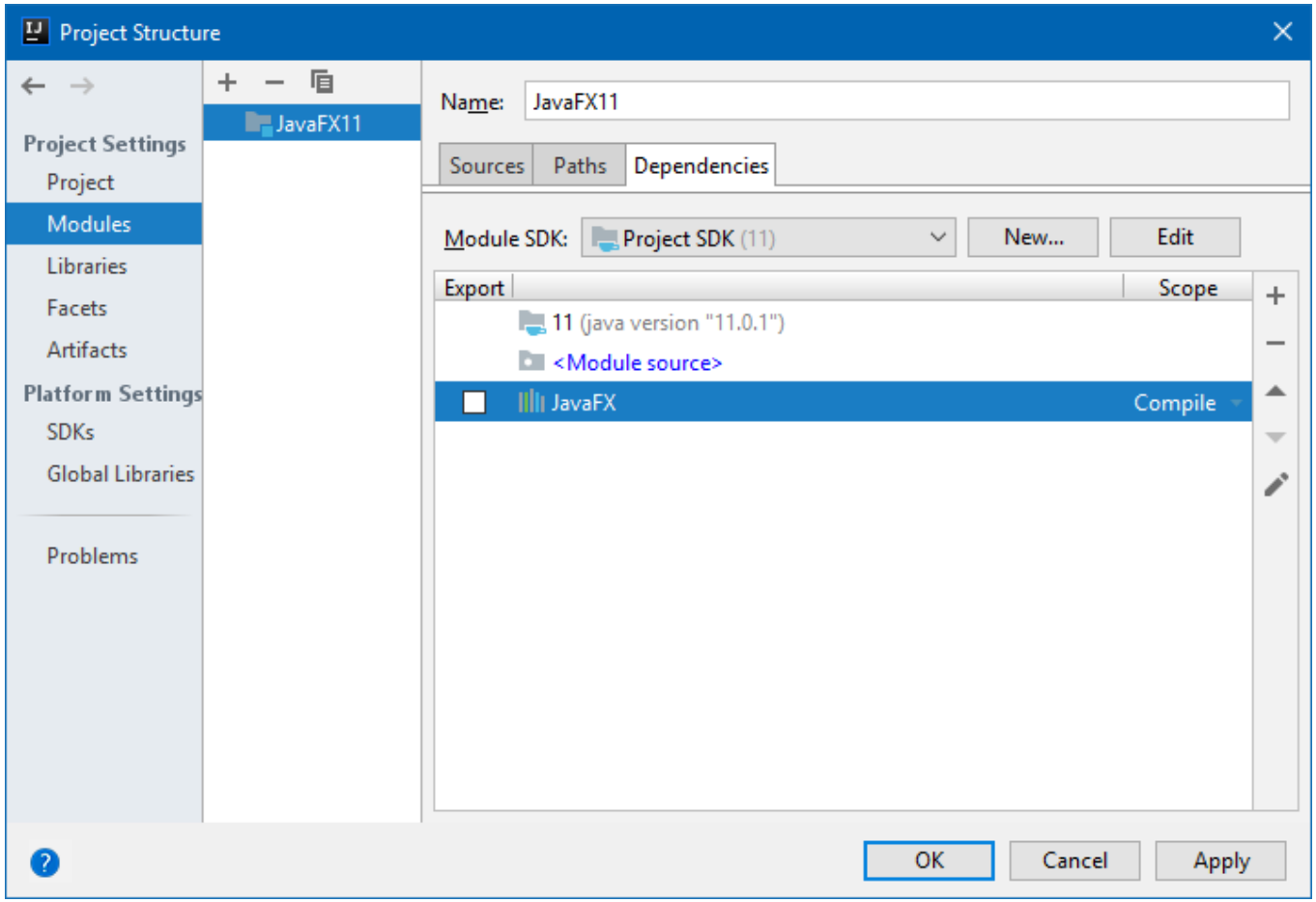
Select **Modules**

In the Dependencies tab click + *(on the rightmost location)* **and Select Library**

Among the Global Libraries select the previously create JavaFX library (click Add selected)

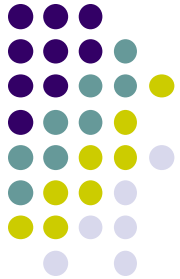
Click OK

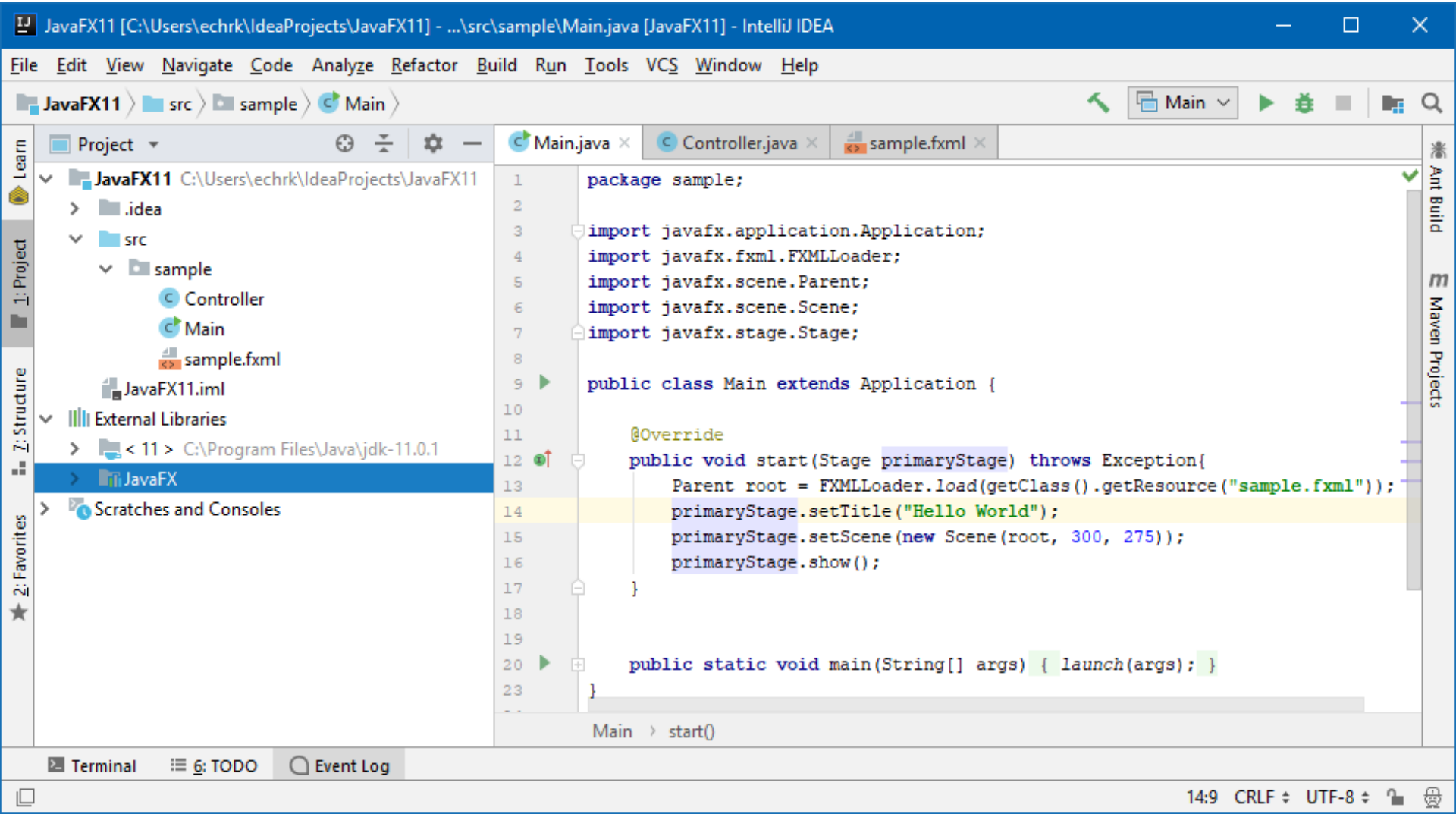
Non-Modular JavaFX project



Non-Modular JavaFX project

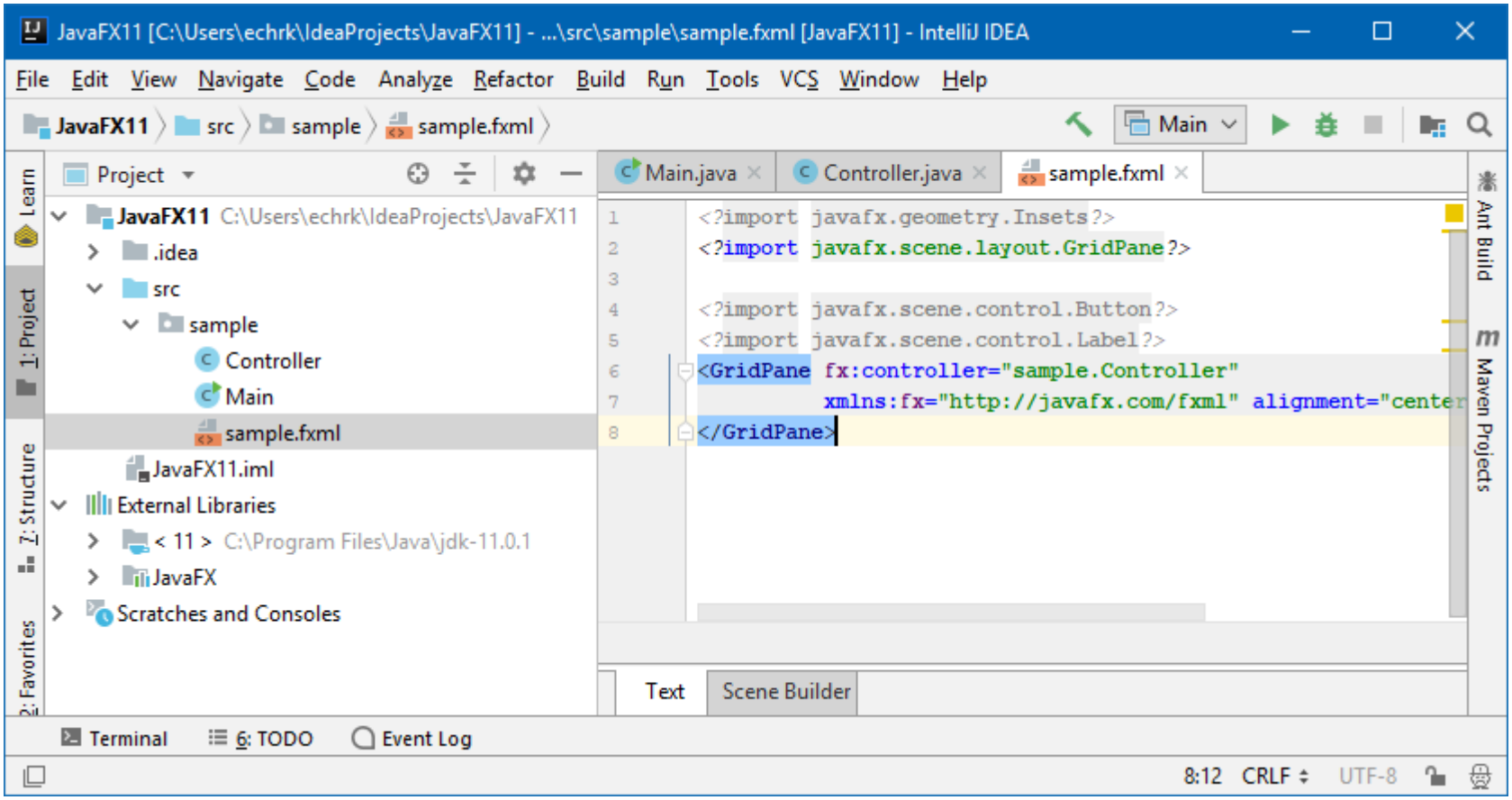
Now you can compile JavaFX 11 source with and JDK 11





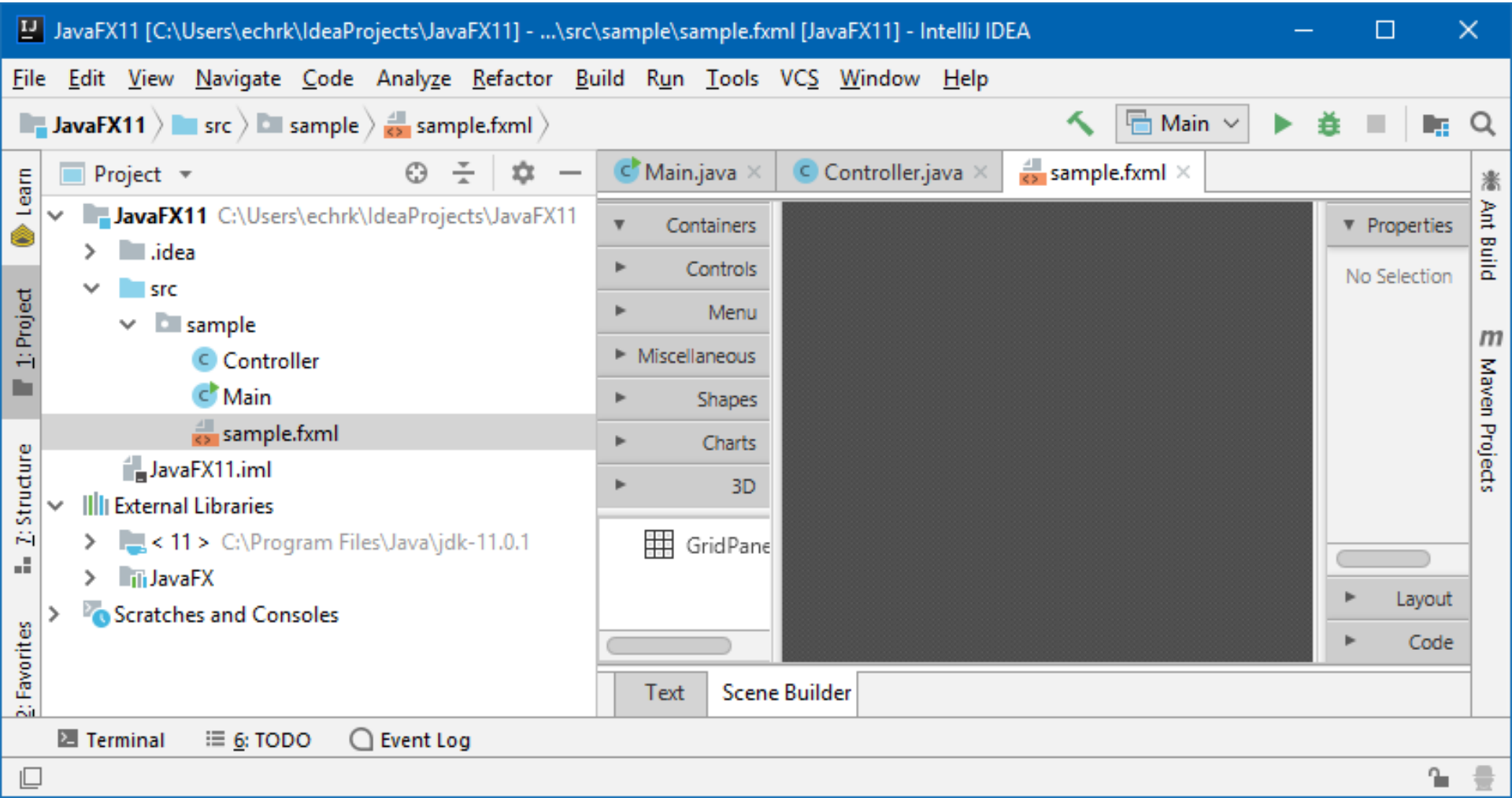
Non-Modular JavaFX project

Select the file (FXML) of the Scene and click the Tab SceneBuilder to edit the Scene with SceneBuilder

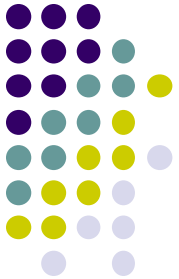


Non-Modular JavaFX project

Edit the Scene with SceneBuilder



Non-Modular JavaFX project



Warning: If you run now the project it will compile but you will get this error:

Error: JavaFX runtime components are missing, and are required to run this application

This error is shown since the **Java 11** launcher checks if the main class extends **`javafx.application.Application`**. If that is the case, it is required to have to **add** the **`javafx.graphics`** module on the **module-path**.

4. Add VM options to resolve the problem

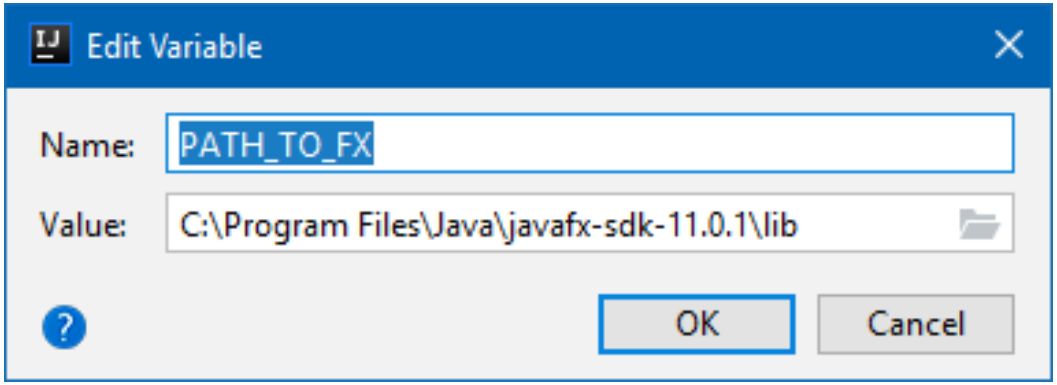
```
--module-path "C:\Program Files\Java\javafx-sdk-11.0.1\lib"
--add-modules=javafx.controls,javafx.fxml
```

Note that the default project created by IntelliJ uses FXML, so **`javafx.fxml`** is required along with **`javafx.controls`**. If your project uses other modules, you will need to add them as well

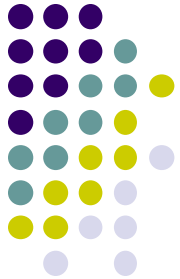
Non-Modular JavaFX project

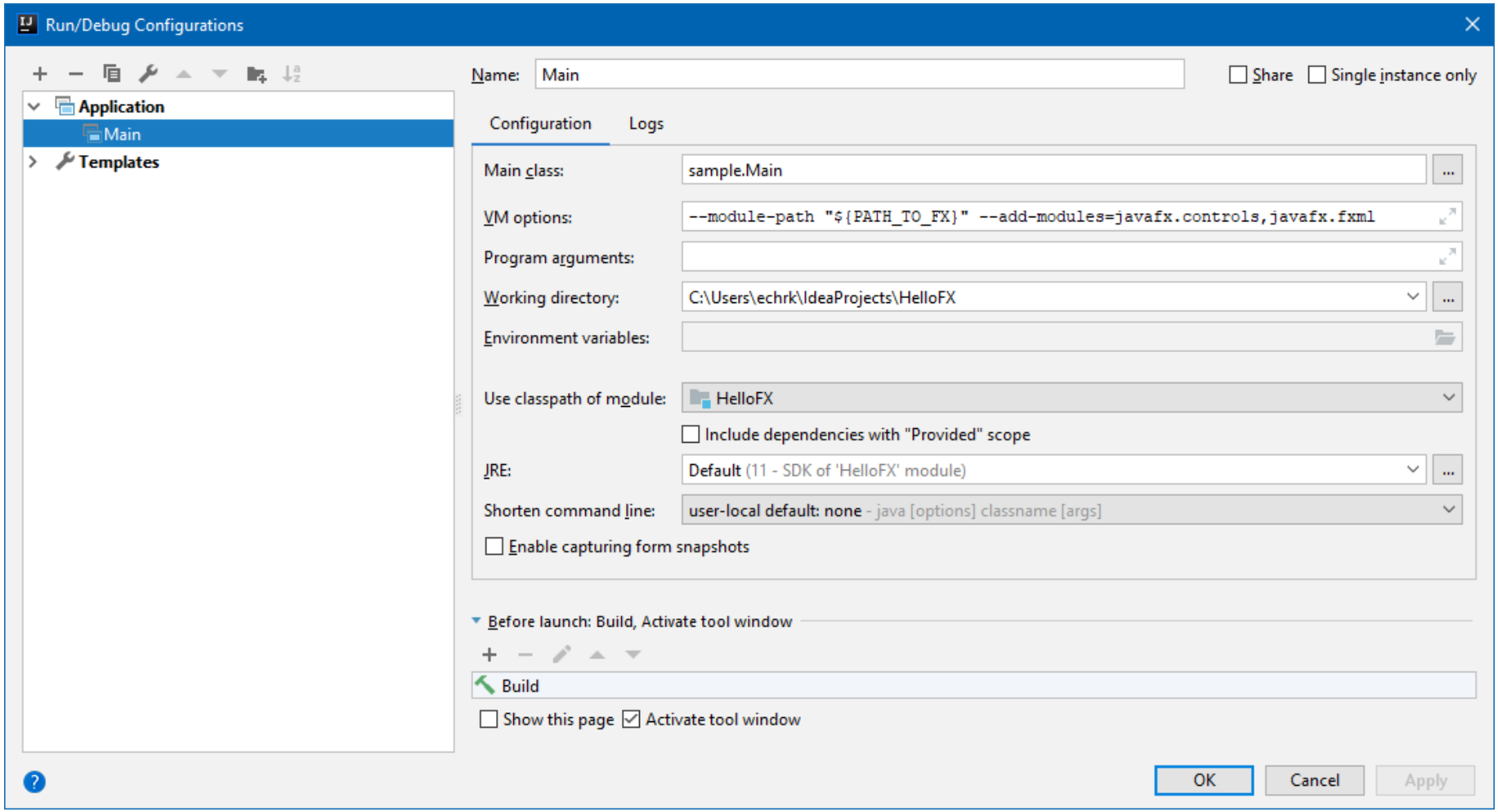
Alternatively, you can define a **global variable** that can be used in future projects.

Go to **File -> Settings** -> **Appearance & Behavior** -> **Path Variables**, and define the name of the variable as **PATH_TO_FX**, and browse to the **lib** folder of the JavaFX SDK to set its value, and click **Apply**

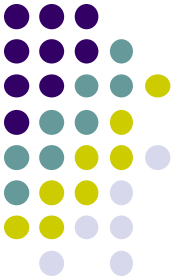


Non-Modular JavaFX project

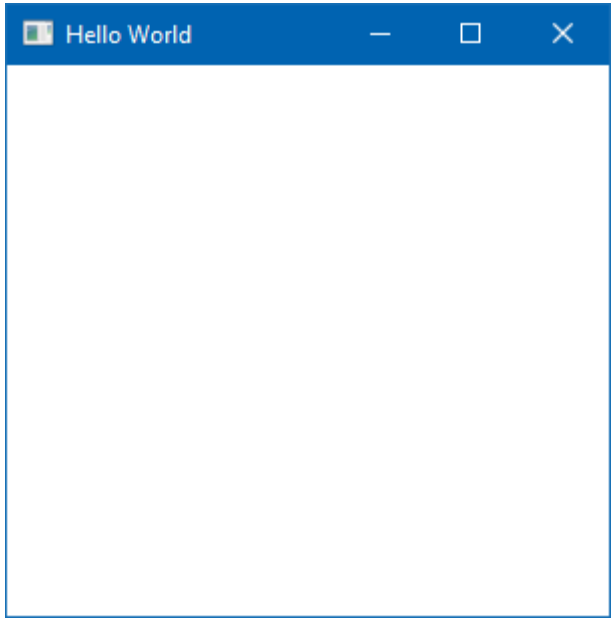


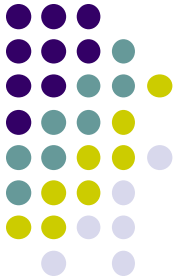


Non-Modular JavaFX project



Now, Run the JavaFX 11 application and see the default window





Happy Object Oriented Programming with JavaFX 11