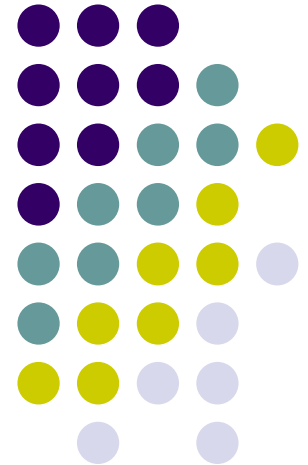


# Setup JavaFX with JDK 13

## Downloads

[JDK 13](#) [Documentation](#)

[JavaFX Windows SDK](#) [SceneBuilder](#)





# IntelliJ setup

Download the appropriate [JavaFX SDK](#) for your operating system and unzip it to a desired location, for instance

C:\Program Files\Java\javafx-sdk-13



# 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 13**.
- In the dialog that opens, select the installation directory of the **JDK** to be used and click OK (C:\Program Files\Java\jdk-13)



# 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:\Program Files\SceneBuilder**

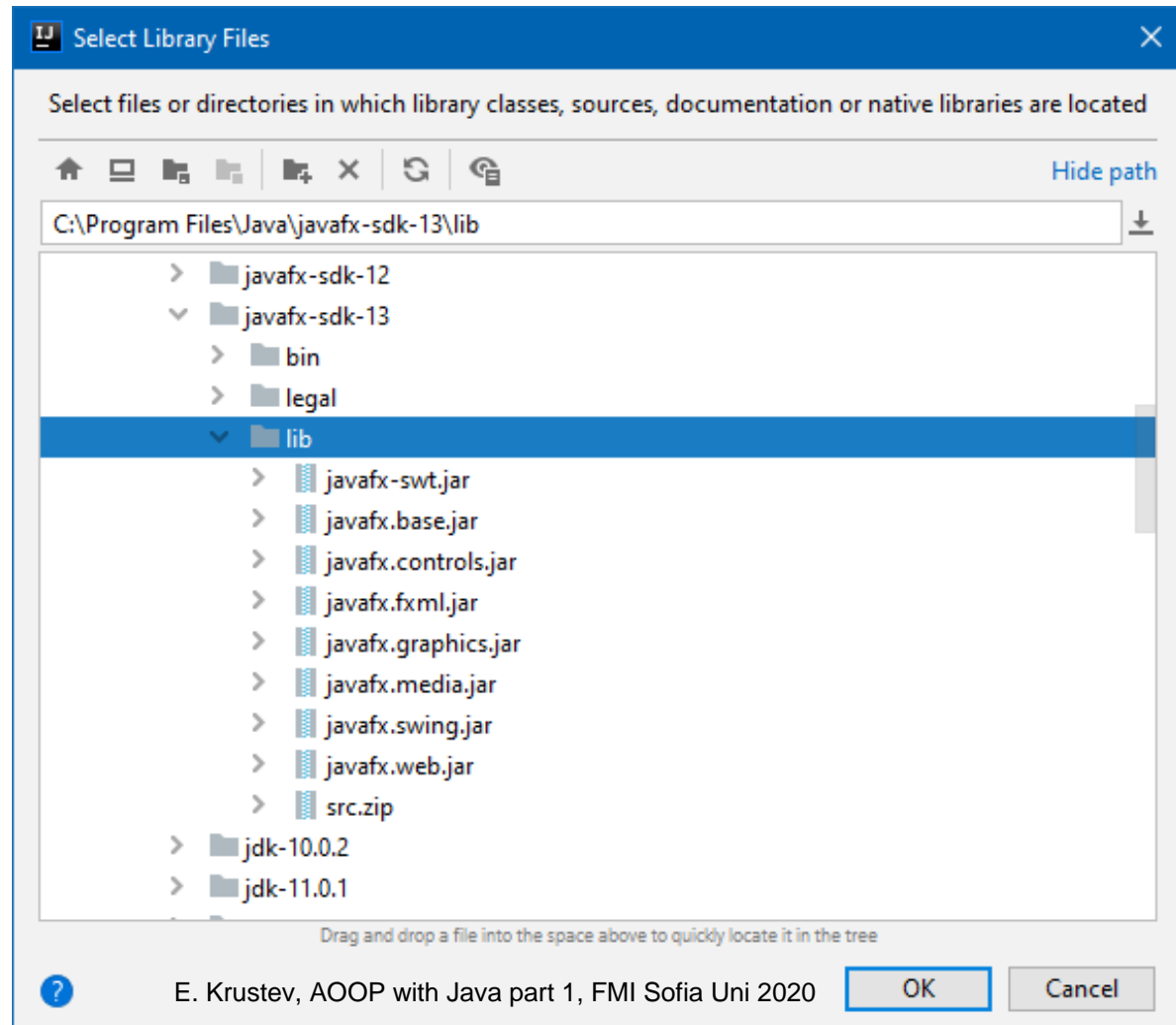


# IntelliJ setup

## Setup JavaFX with JDK 13 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-13\lib).**

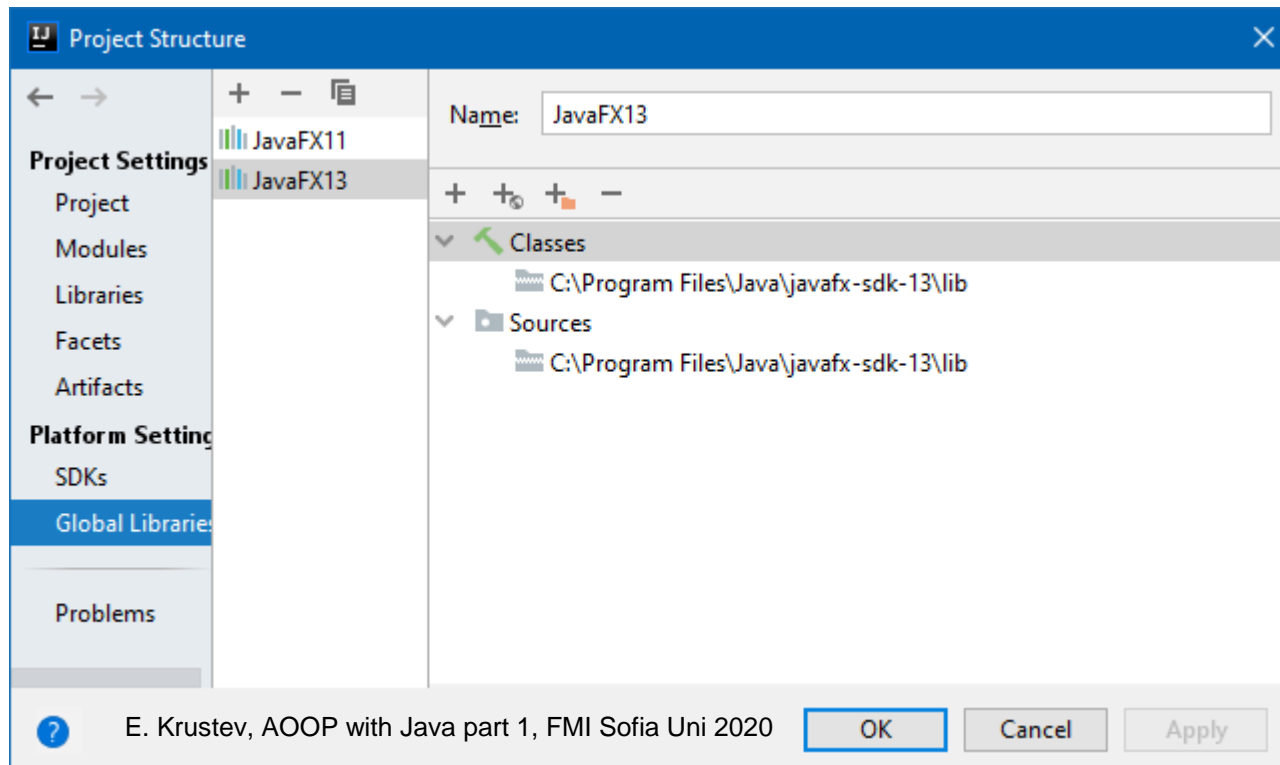
# IntelliJ setup

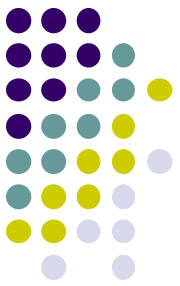




# IntelliJ setup

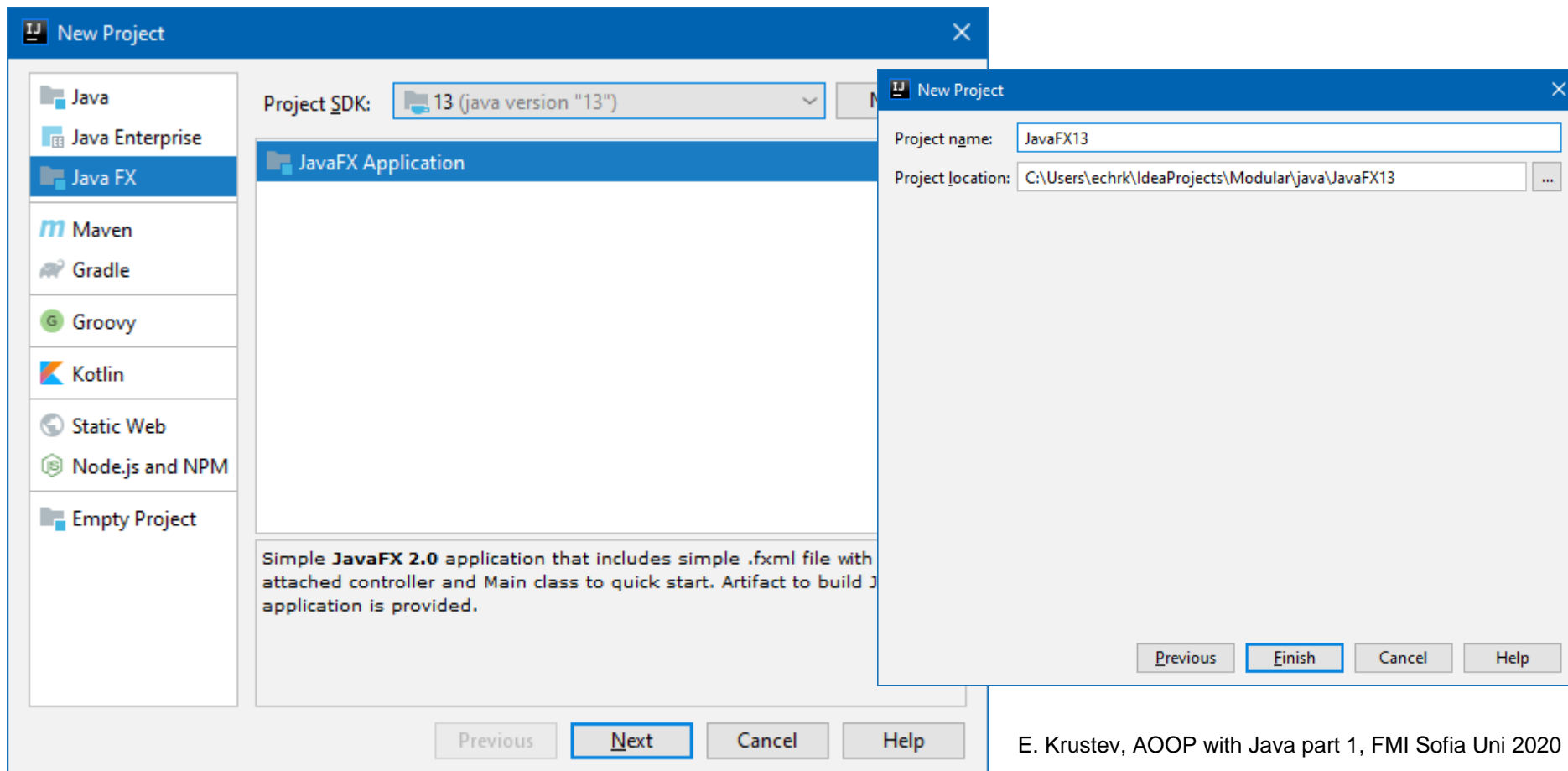
Assign a descriptive name for the Global library, for example **JavaFX13**





# Non-Modular JavaFX project

Create a JavaFX project in IntelliJ in JDK 13.  
Use JDK 13







# Non-Modular JavaFX project

Initially JavaFX 13 is not recognized

```

Main.java x Controller.java x sample.fxml x
1 package sample;
2
3 import javafx.application.Application;
4 import javafx.fxml.FXMLLoader;
5 import javafx.scene.Parent;
6 import javafx.scene.Scene;
7 import javafx.stage.Stage;
8
9 public class Main extends Application {
10
11     @Override
12     public void start(Stage primaryStage) throws Exception{
13         Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));
14         primaryStage.setTitle("Hello World");
15         primaryStage.setScene(new Scene(root, 300, 275));
16         primaryStage.show();
17     }
18
19
20     public static void main(String[] args) { launch(args); }
21
22 }
23
24
```



# Non-Modular JavaFX project

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

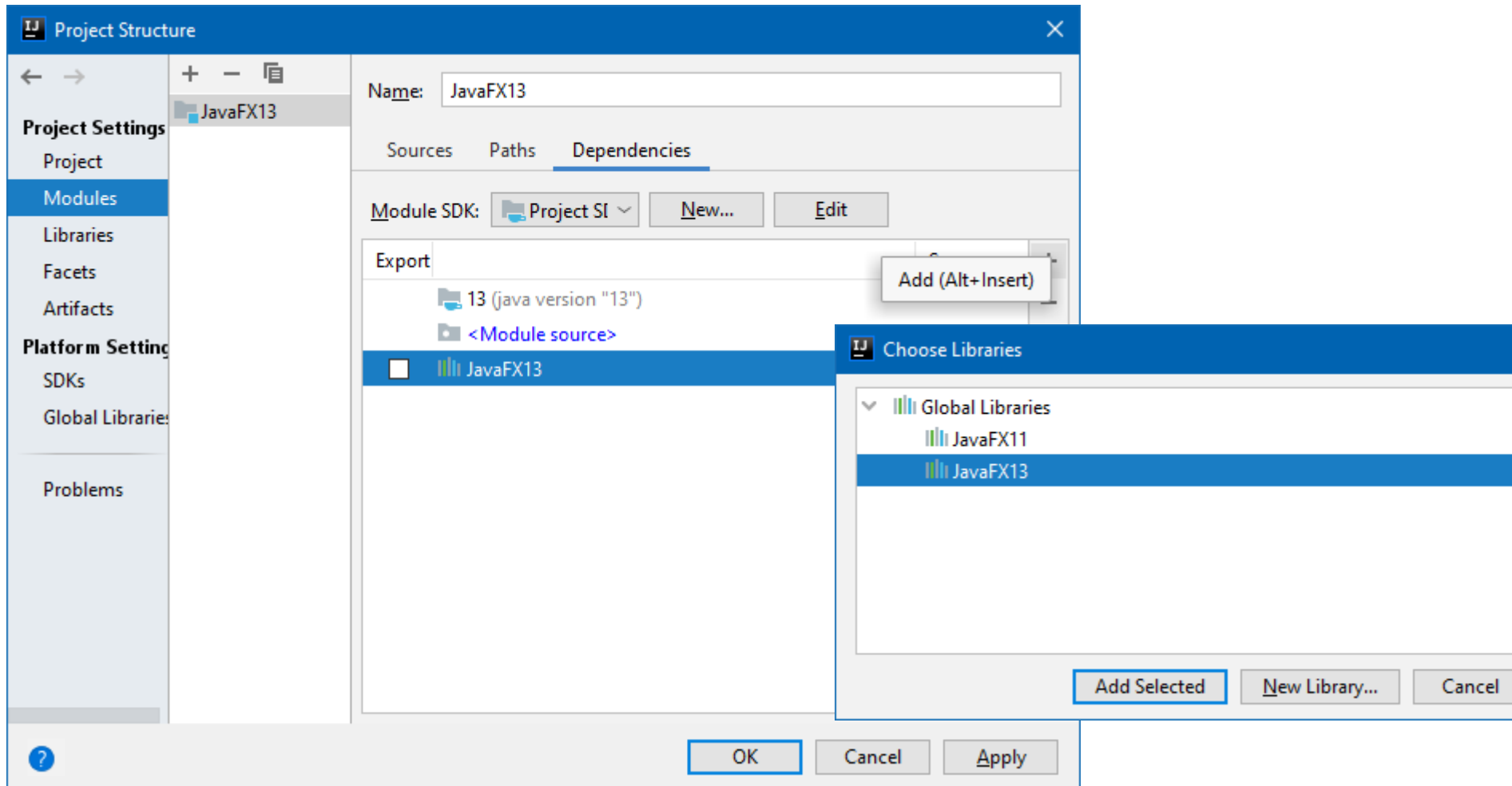
Select **Modules**

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**)

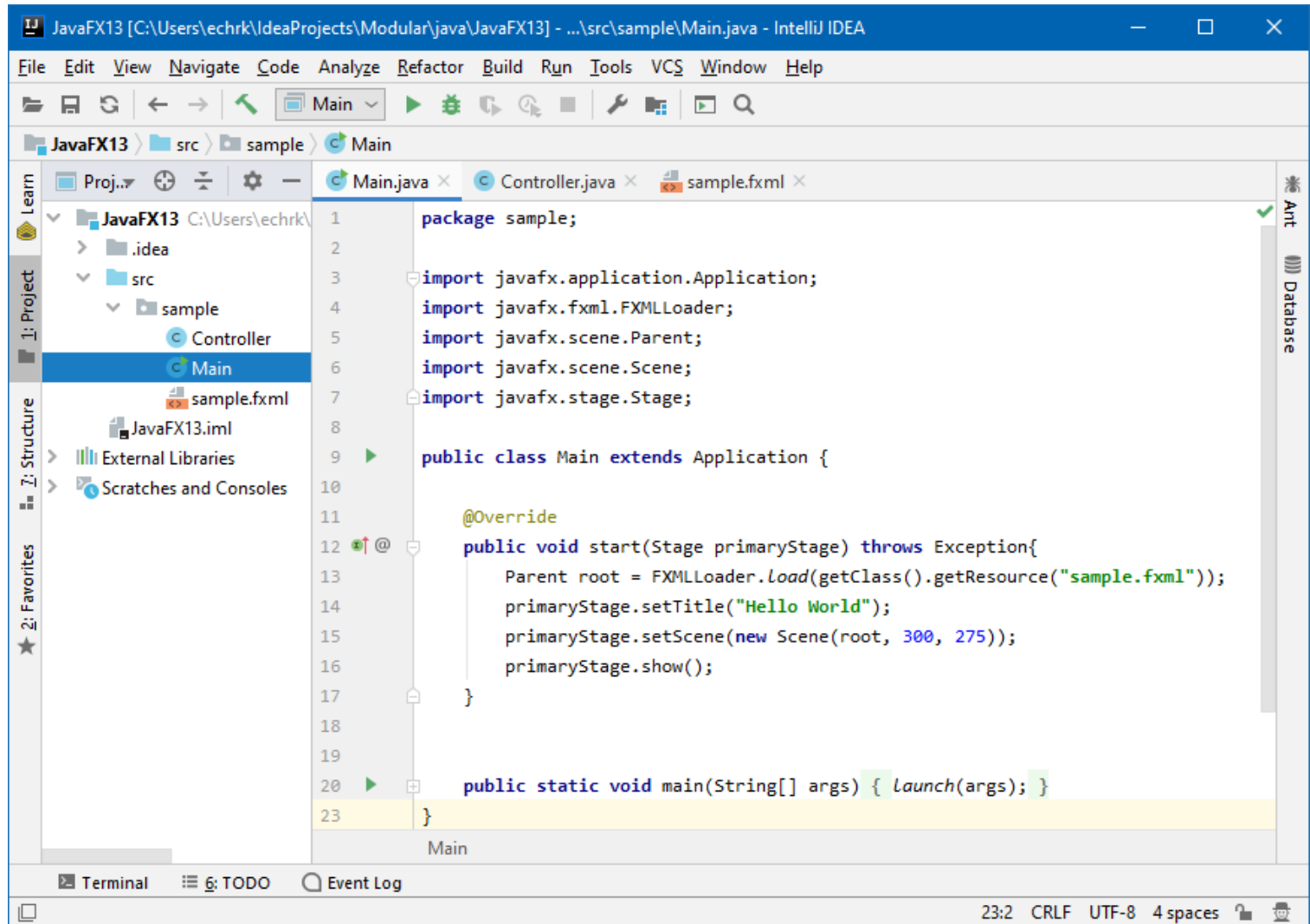
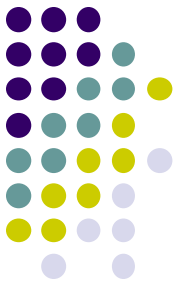
Click **OK**

# Non-Modular JavaFX project

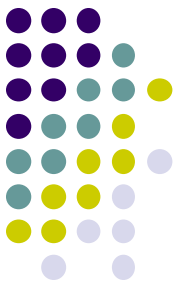


# Non-Modular JavaFX project

Now you can compile JavaFX 13 source with and JDK 13

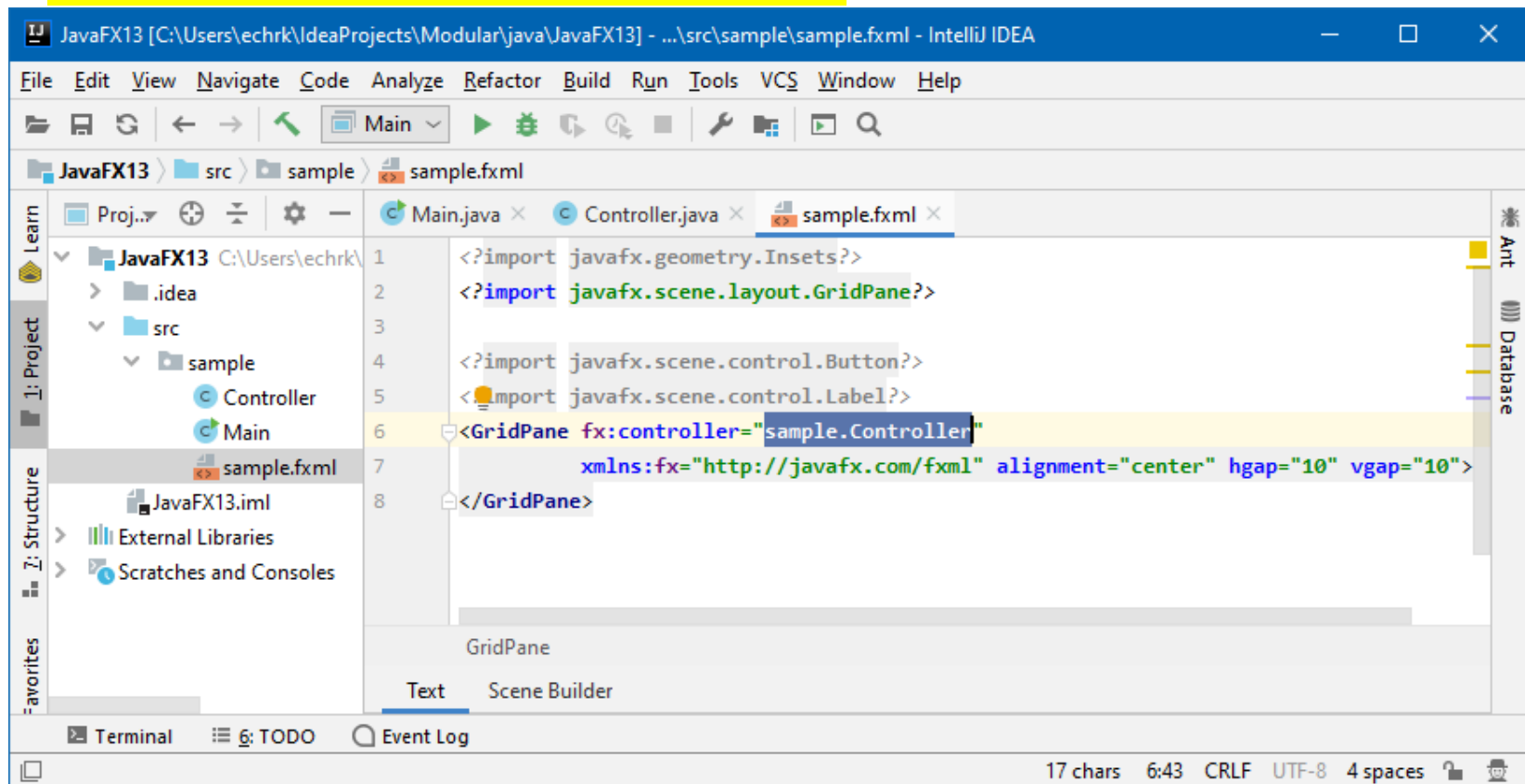


# Non-Modular JavaFX project



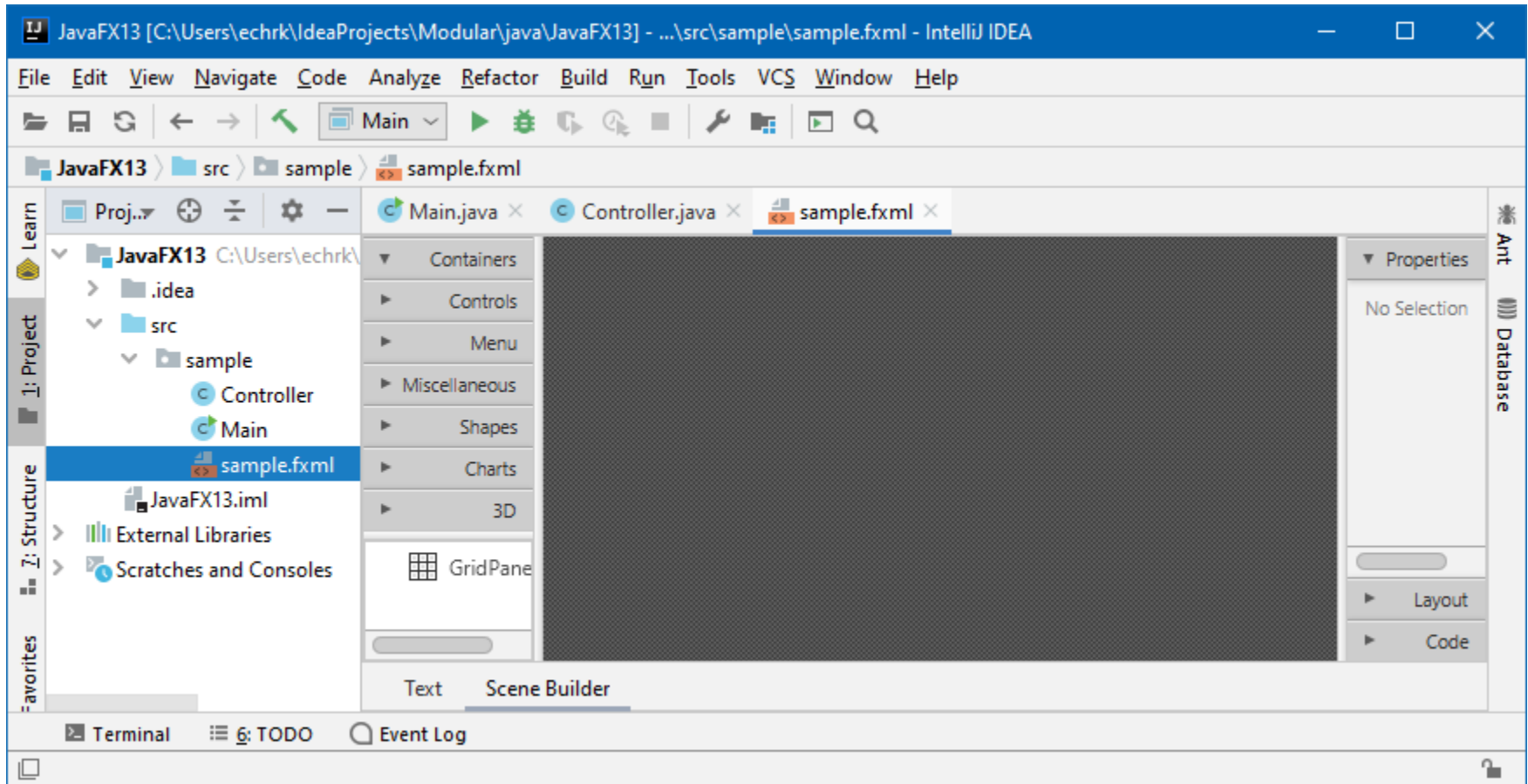
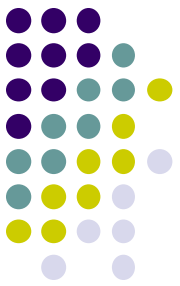
Select the file (FXML) of the Scene and click the Tab **SceneBuilder** to edit the Scene with SceneBuilder or Right click it to select **Open in Scene Builder (better!)**

**Note:** `fx:controller` must be the name of `sample.Controller.java` (incl. package name)

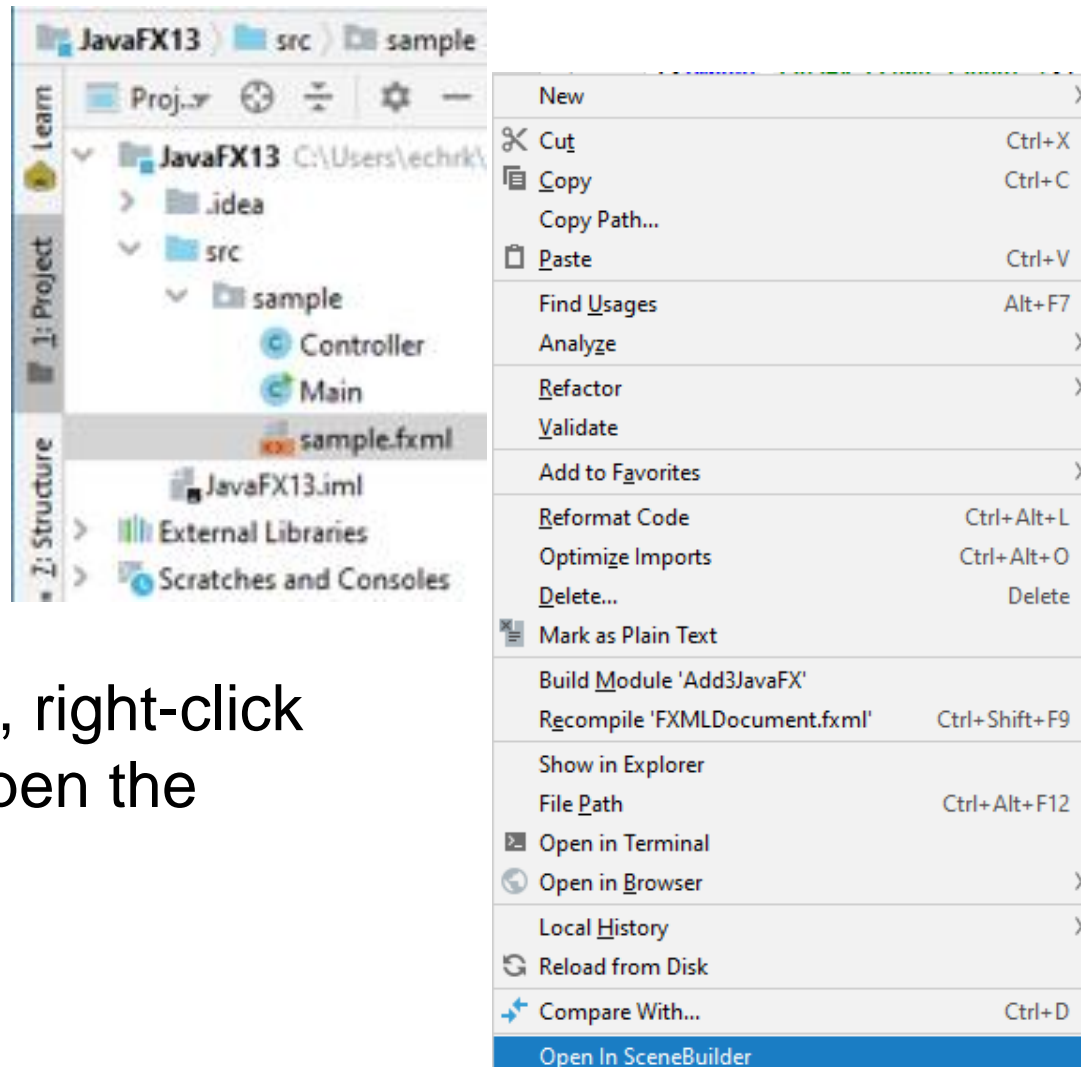
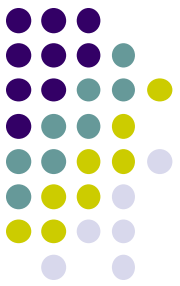


# Non-Modular JavaFX project

## Edit the Scene with SceneBuilder



# Non-Modular JavaFX project



**Note:** Preferably, right-click the FXML and open the Scene in Scene Builder

# 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 13** 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-13\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

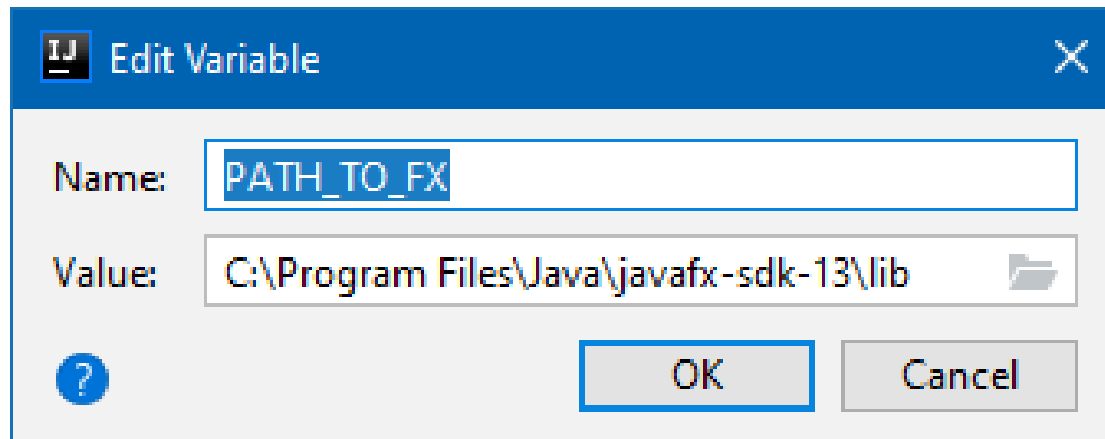
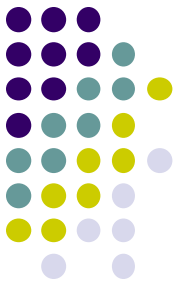


# Non-Modular JavaFX project

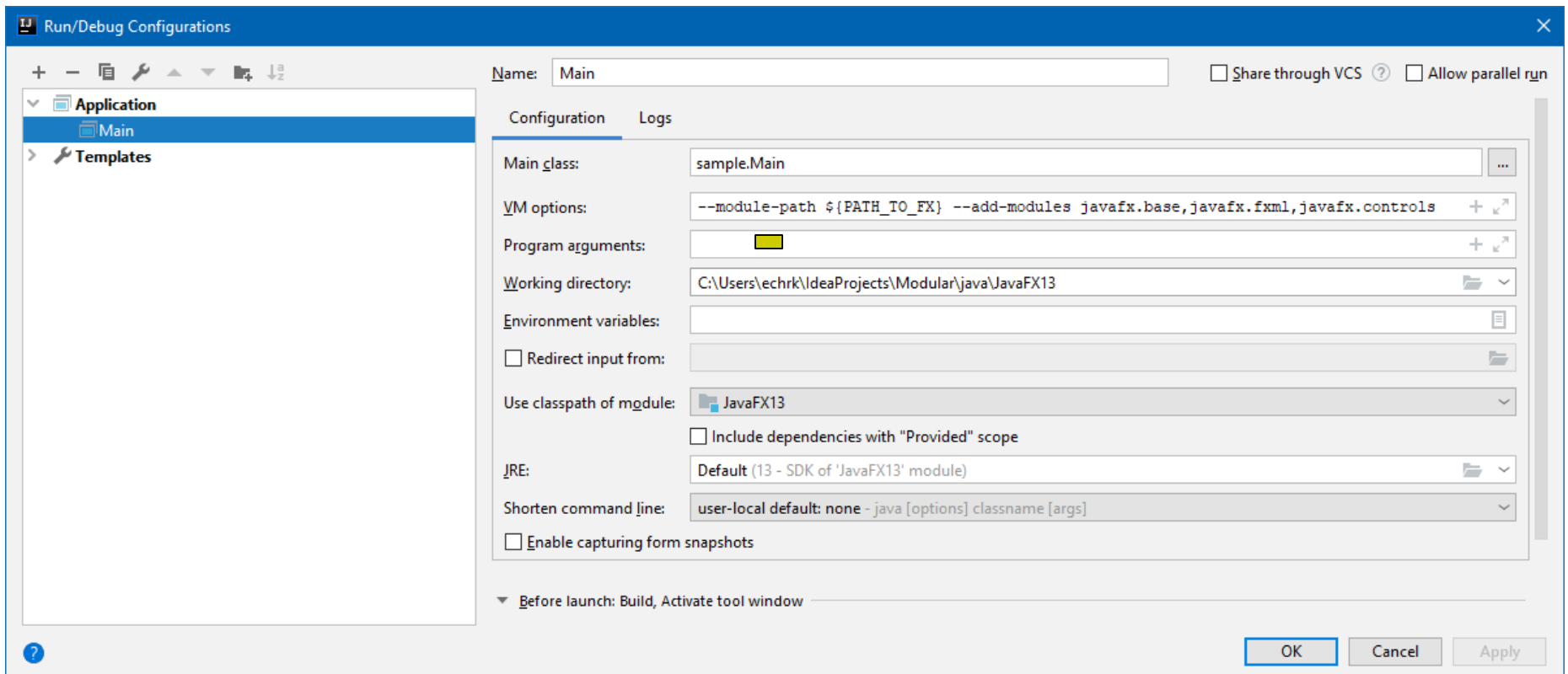
Alternatively, you can define a **global variable** that can be used in future projects.

Go to **File -> Settings -> Appearance & Behavior -> Path Variables**,

and define the name of the variable such as **PATH\_TO\_FX**, and browse to the **lib** folder of the JavaFX SDK to set its value, and click **Apply**



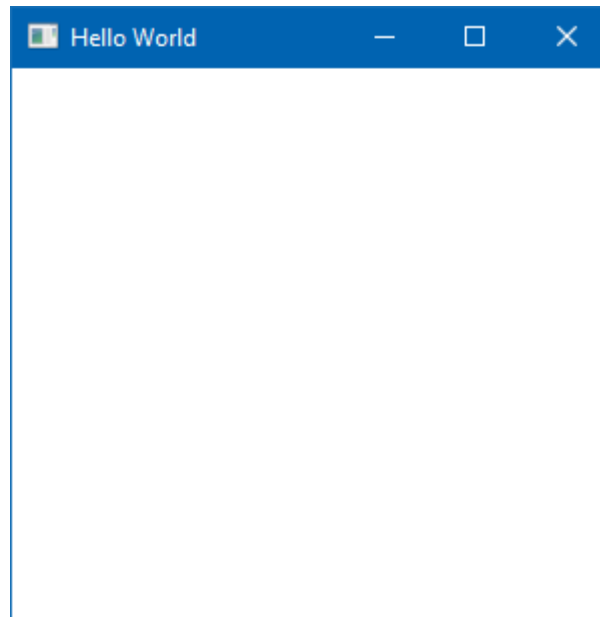
# Non-Modular JavaFX project



# Non-Modular JavaFX project



Now, Run the JavaFX 13 application and see the default window





# **Happy Object Oriented Programming with JavaFX 11+**