Setup JavaFX with JDK 13

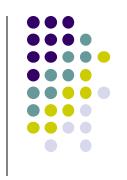
Downloads

JDK 13 Documentation

JavaFX Windows SDK SceneBuilder







Download the appropriate <u>JavaFX SDK</u> for your operating system and unzip it to a desired location, for instance

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



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)





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





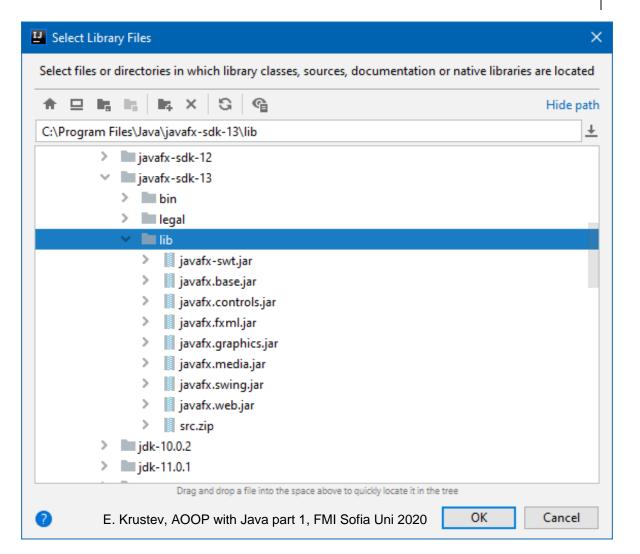
Setup JavaFX with JDK 13 as a Global library

- 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).

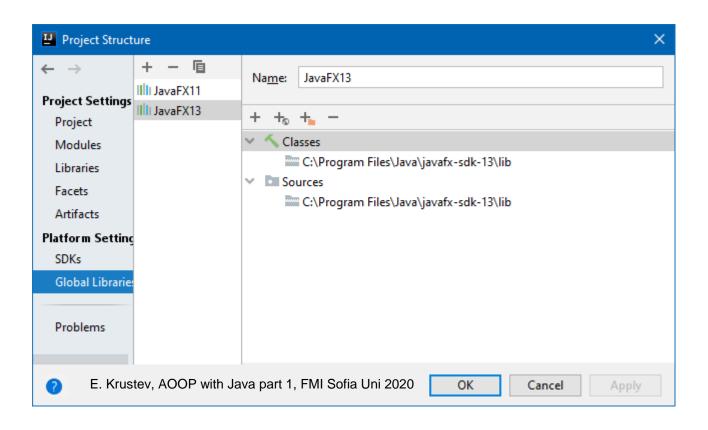
IntelliJ setup





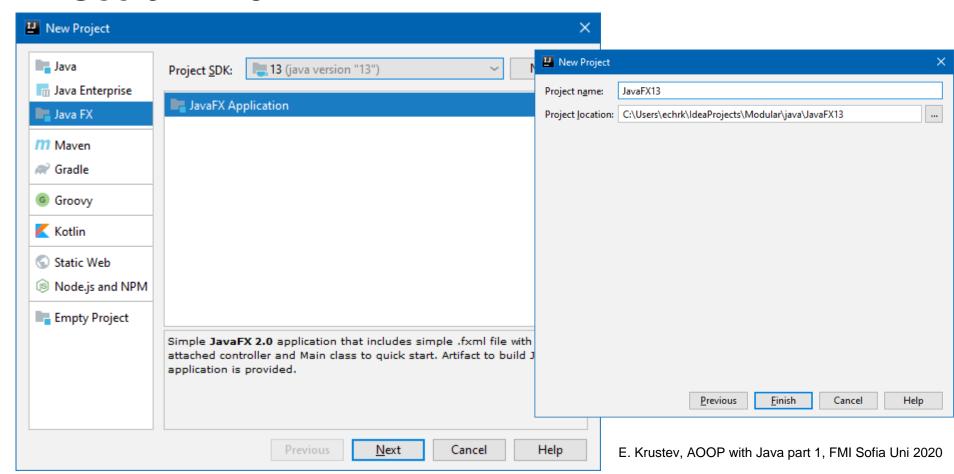


Assign a descriptive name for the Global library, for example **JavaFX13**





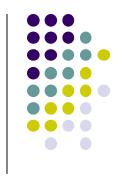
Create a JavaFX project in IntelliJ in JDK 13. Use JDK 13





Initially JavaFX 13 is not recognized

```
package sample;
       import javafx.application.Application;
       import javafx.fxml.FXMLLoader;
       import javafx.scene.Parent;
       import javafx.scene.Scene;
       import javafx.stage.Stage;
8
       public class Main extends Application {
9
10
11
           @Override
           public void start(Stage primaryStage) throws Exception{
12
               Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));
13
               primaryStage.setTitle("Hello World");
14
               primaryStage.setScene(new Scene(root, 300, 275));
15
16
               primaryStage.show();
17
18
19
           public static void main(String[] args) { launch(args); }
20
23
                     E. Krustev, AOOP with Java part 1, FMI Sofia Uni 2020
24
```



Select File->Project Structure->Project structure

Select Modules

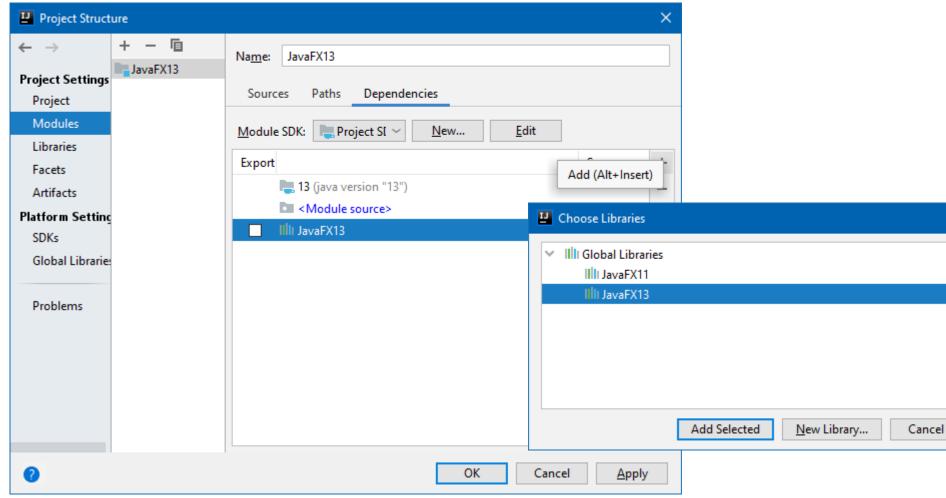
In the Dependencies tab click + (on the rightmost location) and Select Library

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

Click OK

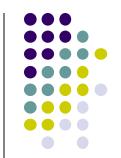






E. Krustev, AOOP with Java part 1, FMI Sofia Uni 2020

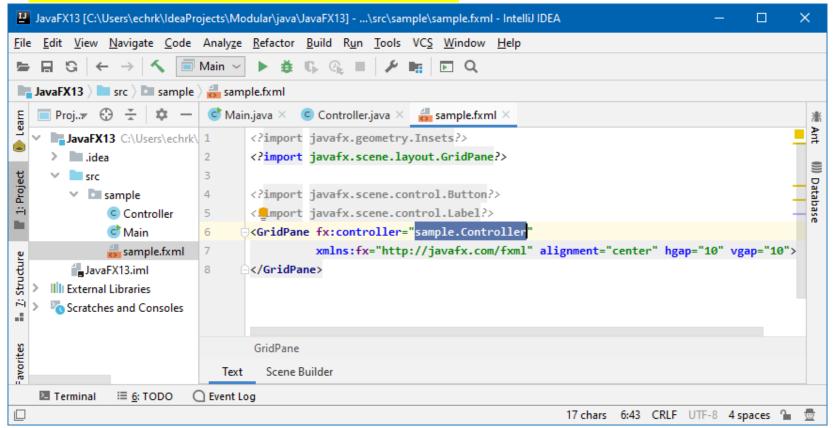
Now you can compile JavaFX 13 source with and JDK 13



```
JavaFX13 [C:\Users\echrk\IdeaProjects\Modular\java\JavaFX13] - ...\src\sample\Main.java - IntelliJ IDEA
                                                                                                                    File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
                            Main
JavaFX13 > src > sample > d Main
      Proj.. T
                              Main.java ×
                                            Controller.java ×
                                                               JavaFX13 C:\Users\echrk\
                                        package sample;
     > idea
                              3
                                        import javafx.application.Application;
1: Project
        src
        sample
                                        import javafx.fxml.FXMLLoader;
               Controller
                              5
                                        import javafx.scene.Parent;
                              6
                                        import javafx.scene.Scene;
               Main
              ample.fxml
                              7
                                        import javafx.stage.Stage;
Structure
                              8
         JavaFX13.iml
     III External Libraries
                              9
                                        public class Main extends Application {
69
      Scratches and Consoles
                              10
                              11
                                            @Override
                              12 💵 @
                                            public void start(Stage primaryStage) throws Exception{
Favorites
                              13
                                                Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));
                                                primaryStage.setTitle("Hello World");
                              14
čij
                              15
                                                primaryStage.setScene(new Scene(root, 300, 275));
                              16
                                                primaryStage.show();
                              18
                              19
                                            public static void main(String[] args) { launch(args); ]
                              23
                                        Main
   Terminal
                Event Log
23:2 CRLF UTF-8 4 spaces 🚡
```

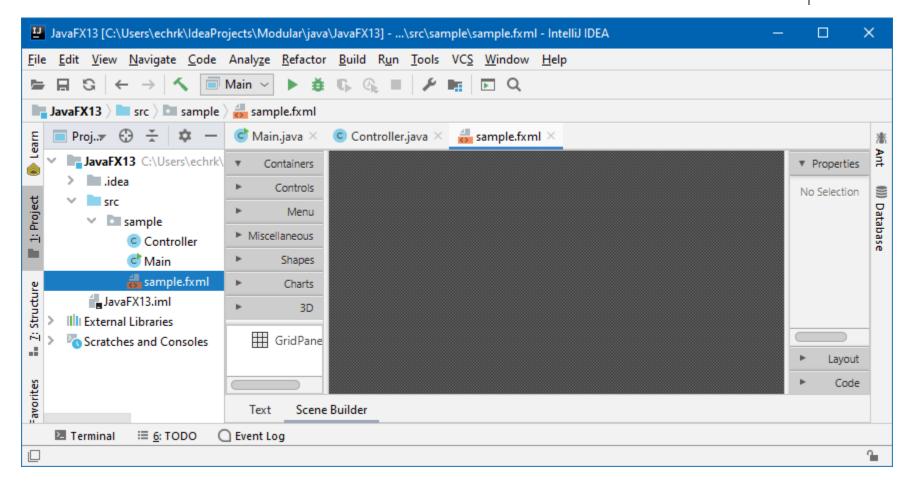
Select the file (FXML) of the Scene and click the Tab **SceneBuilder** to edit the Scene with SceneBuilder or Right click it to select **Open in Scene Builder (better!)**

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

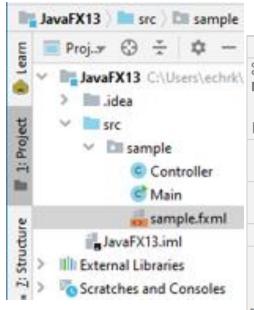


Edit the Scene with SceneBuilder

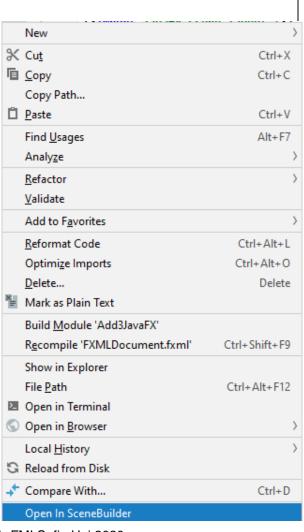








Note: Preferably, right-click the FXML and open the Scene in Scene Builder



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 <code>javafx.application.Application</code>. If that is the case, it is required to have to add the <code>javafx.graphics</code> 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
```

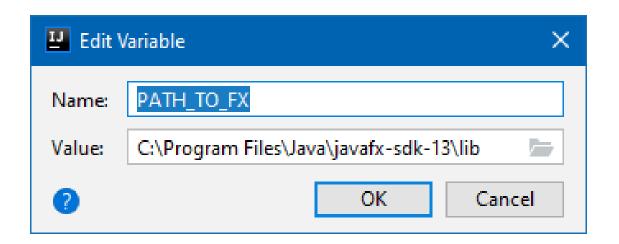
Note that the default project created by IntelliJ uses FXML, so javafx.fxml is required along with javafx.controls. If your project uses other modules, you will need to add them as well

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**

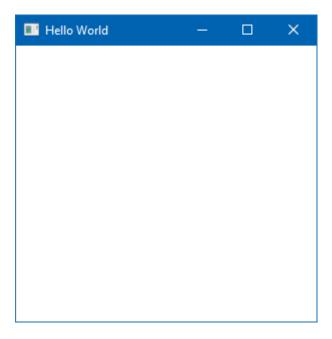




Run/Debug Configurations				×
+ - □	Name: Main Configuration Logs	☐ <u>S</u> hare through VCS ⑦ ☐ Allo	ow parallel r	<u>u</u> n
> F Templates	Main <u>c</u> lass:	sample.Main		
	<u>V</u> M options:	module-path \${PATH_TO_FX}add-modules javafx.base,javafx.fxml,javafx.controls	+ 🖟	
	Program a <u>rg</u> uments:		+ 🗷	
	Working directory:	C:\Users\echrk\ldeaProjects\Modular\java\JavaFX13	- ~	
	Environment variables:			
	Redirect input from:			
	Use classpath of module:	JavaFX13	~	
		☐ Include dependencies with "Provided" scope		
	<u>J</u> RE:	Default (13 - SDK of 'JavaFX13' module)	- ×	
	Shorten command <u>l</u> ine:	user-local default: none - java [options] classname [args]	~	
	Enable capturing form	snapshots		
	▼ <u>B</u> efore launch: Build, Acti	vate tool window		
②		OK Cancel	Apply	



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





Happy Object Oriented Programming with

JavaFX 11+