Setting up JavaFX

It's GUI time!
We're now at a place where we're going to need to work with Graphical User Interfaces, or GUIs. A **GUI** gives an application its “look and feel”. GUIs are typically built of **GUI components** from GUI toolboxes; components are often called “**widgets**”, and **JavaFX is a framework** that contains many classes of GUI components. So let's get it installed!

What tools will you need?

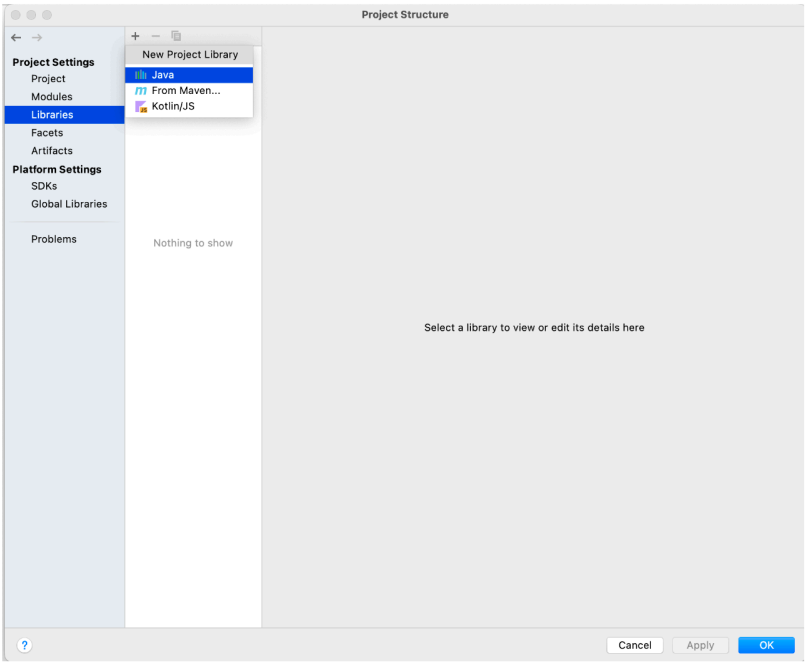
To get started, you're going to need to download the latest version of a graphics library called JavaFX. ; API documentation can be found at this link. You can download the version for Java 19 even if you are using Java 18 for your coursework. When you download, you will download an archive; make sure to decompress these files and store them in some location you can remember. On a Mac, you might want to store java libraries in your "/Users/Username/Library" folder. On a Windows PC, you might want to store your libraries in your "C:\Program Files\Java" folder.

Configuring a JavaFX Project

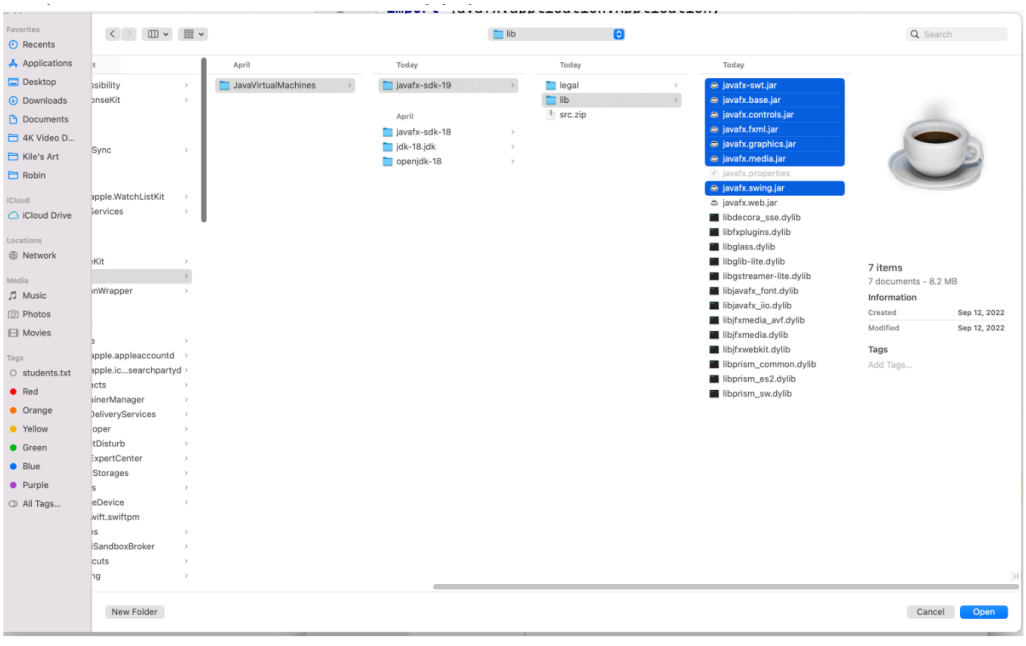
Now we have to tell IntelliJ where your JavaFX library is located. You can start by opening a folder with a project, like the Assignment 2 starter code or the Lab 05 starter code, within IntelliJ. You'll need to tell IntelliJ which version of Java you intend to use, and to add JUnit5 to your class path, as always. But even after you do that, you'll note any lines in your code that reference JavaFX will be highlighted in red, as illustrated below.

```
import javafx.application.Application;
import javafx.collections.FXCollections;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.ChoiceBox;
import javafx.scene.control.TextField;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.AnchorPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Priority;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.stage.Stage;
import javafx.geometry.Insets;
```

To fix this, we need to add the JavaFX Library to your project. You can to this by navigating to Project Structure -> Libraries -> Global Libraries. Click on the '+' button at the top of the screen and select "Java" as illustrated below.



Then navigate to the folder in which you placed your JavaFX files. Locate the subfolder named "lib" that likes within the decompressed archive, and select all of the files ending in ".jar" to add to this library, as illustrated below. Name your Library something like "javafx" so you can easily find it again when you next need it. Click on "Apply" and then "Close".



This should make all of the red highlights related to JavaFX in your code disappear. You should also now be able to compile your starter code. When you try to "run", however, you'll likely encounter another error, which is illustrated below. This time the error should read something like this:

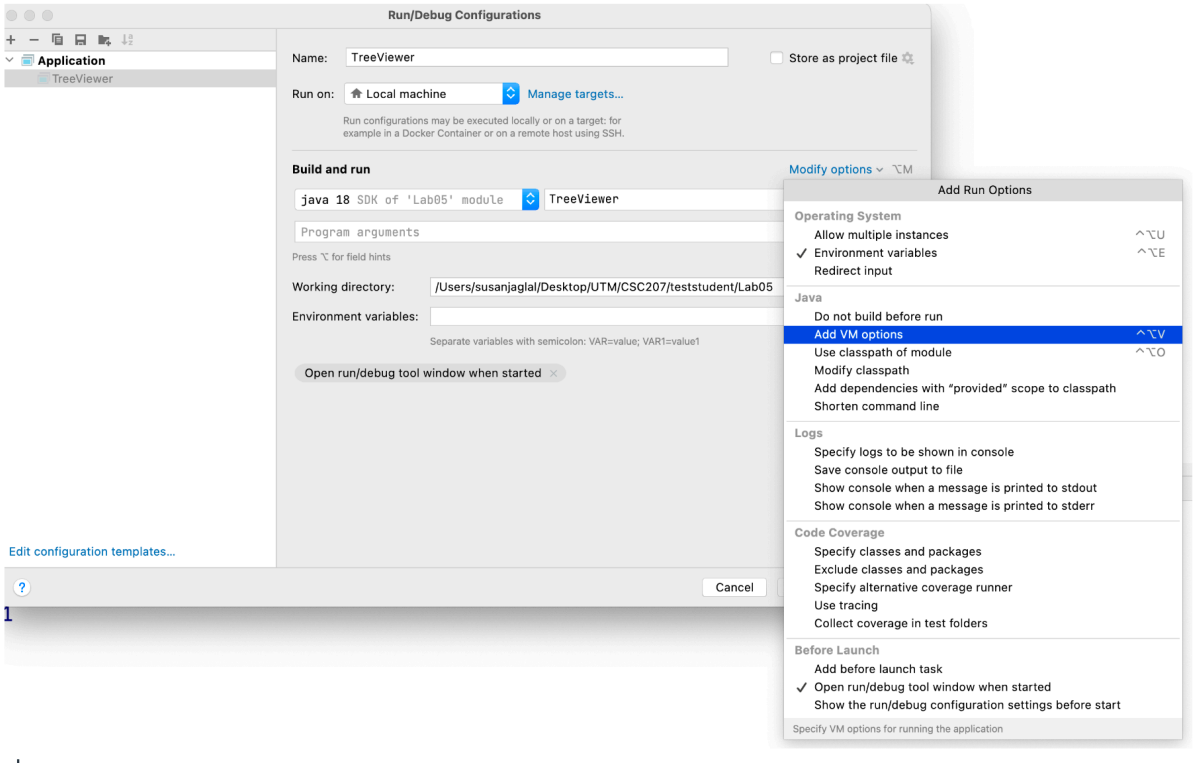
Error: JavaFX runtime components are missing, and are required to run this application

To fix this problem, we need to add "Virtual Machine" or VM arguments to your run settings. To do this, navigate to Run -> Edit Configurations. Within your Run/Debug configuration, select Modify Options -> add VM options. Add the following line to your VM arguments:

```
--module-path "path/to/javaFX/" --add-modules javafx.controls,javafx.fxml
```

And make sure you change the path (in bold above) to the directory where JavaFX library was saved to your machine.

This is illustrated below.



Click "Apply" followed by "OK" ... and you should be good to go!

Much of this process is illustrated in the following videos:

A video tutorial of lab set up:

https://web.microsoftstream.com/video/b5e1ed89-0c5a-4da1-927c-4b71b067ca7d

A video of the lab version that includes javax.accessibility screen reader hooks:

https://web.microsoftstream.com/video/9713e085-1112-478d-a062-9ba214b43726