LAB 3: SIMPLE GUI DEVELOPMENT WITH JAVAFX and SWING

Aims:

- Ensure that JavaFX and SWING work on your computer
- Gain some GUI development experience by re-implementing Lecture 03A's examples
- Gain some GUI development experience by building a simple phone app in SWING and JAVAFX.

Please do the exercise(s) that you feel are appropriate for you.

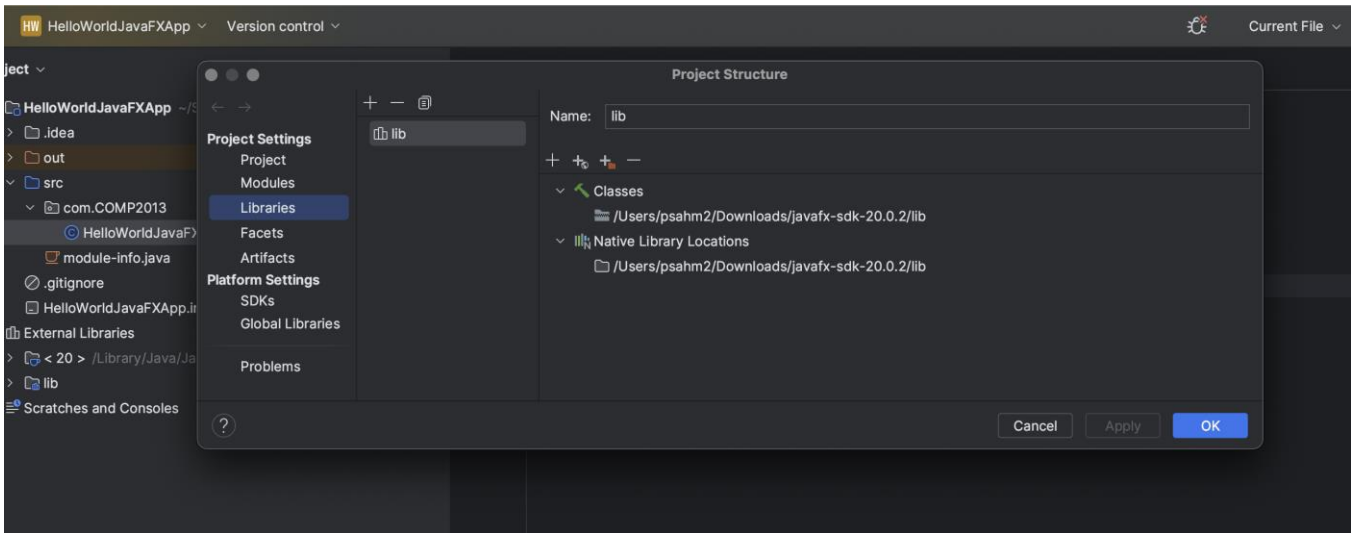
BEGINNER LEVEL: CREATING A JAVAFX PROJECT

To ensure that JavaFX works on your computer, set up some test projects in IntelliJ. Start by doing it manually (as I did in the lecture, using a plain Java project) and then by using the JavaFX project option, which will configure some build script to set up the project and dependencies automatically.

Part 1: RE-IMPLEMENTING and MODIFYING THE LECTURE EXAMPLE in JAVAFX and SWING

Re-implement the HelloJavaFXWorld example presented in Lecture 03A. Then do some modifications to it (e.g. adding components or using different layout panes) and check out the consequences.

There are many ways to set up your JavaFX project.
You need to make sure that you have JavaFX into your library, and you can add javafx to your module-info file just like you have seen in the clas. If this does not work, you can add this manually in Project structure - > Libraries -> Then add the JavaFX folder. If you need to download this here https://jdk.java.net/javafx20/



Here is how your module-info file should look like:

```
module HelloWorldJavaFXApp {
    opens com.COMP2013;
    requires javafx.graphics;
    requires javafx.controls;
}
```

Part 2: TELEPHONE APPLICATION in JavaFX

Your main task for the lab is to build a simple telephone application. You can find the spec below.

Specification: You need to build a JavaFX GUI-based app with the following features:

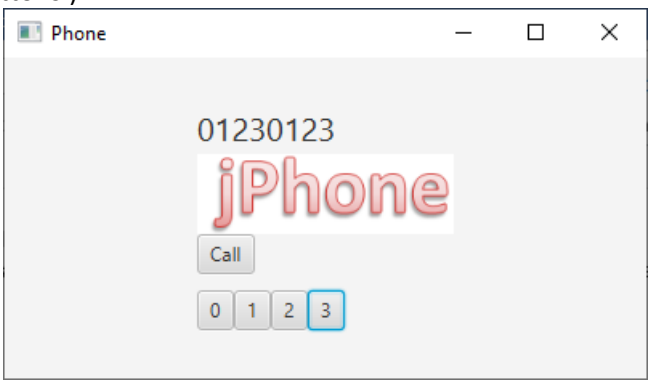
1. Number buttons (0 to 9) to enter the phone number {Tip: A GridPane might be useful here}
2. A display to view the current number
3. A Call button. This will obviously not actually make a call, but should pass the number to a ***makeCall()*** method, which implements an ***Phone*** interface. This class should represent a dummy telephone device, and the method should wait for a random duration (in seconds) and then return the duration of the "call".

```
//Things which can make or fake telephone calls implement this.
public interface Phone {
    public int makeCall(String number);
}
```

   - Note how, by implementing an interface, we can switch out a "dummy" or "real" phone application easily.
   - To wait for some time you could do something like this (note though, that this example snippet does not wait for a random time, and that this might not be the best way to deal with this requirement):

```
duration=3000; // could make this random
try {
    Thread.sleep(duration); // note how this hangs the interface! how can we fix that?
} catch (InterruptedException e) {
    e.printStackTrace();
} // faking phone call time.
```

4. A logo just below the number display.
   - To do this you need to find out how to display an image on a JavaFX window {Tip: Check out the ImageView class}
   - This is how it could look like (of course you can use different layouts to display all 10 number buttons.):



5. Maybe think on how you could make this look pretty 😊

Things to bear in mind:

- You need to handle telephone numbers starting with a 0, so think about your data type!

Part 3: REWRITE THE TELEPHONE APPLICATION in SWING

Now that you have managed to finalise your desing in JavaFX, we want you to build the same program but using Swing. This is related to legacy code; a lot of projects out there are build in Swing, so learning and understanding will be highly beneficial.

- Start from the Hello World Example in the Monday Lecture, and build on from there. This will require some individual research into how to build different parts.
- For those of you not so familiar with Swing, here are some excellent tutorials to follow in Java Swing