CMPT 3035 Assignment 0: Hello World in JavaFX and FXML

Due: Friday, September 13, 2019, 11:55pm

Overview

You will build ‘hello world’ GUIs in JavaFX using code and using FXML. The goal is to get you going and to test your setup of the development environments, and test the handin system.

This assignment is to be done individually; each student will hand in an assignment.

Requirements

Build ‘hello world’ GUI systems as specified below. Both systems will show a label and a button; when the button is pressed, the label will display the text “Hello 3035!”.

Pre-requisite software

- For this assignment you can use the lab computers, but this is also a good opportunity to Install Java 8 and Eclipse on your personal machines. See the Software Installation Help Notes on D2L.

For JavaFX:

- If you have the e(fx)clipse plugin installed in Eclipse, then you should be able to go to “New Project” -> “Other” -> “JavaFX” -> “JavaFX Project”
- Create a root container (such as a BorderPane), a scene to display your root container, set the primaryStage’s scene to be your scene object, and remember to show your primaryStage. This is all done for your if you use the JavaFX Project template that is given from the step above.
- Create a Button object (see https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/Button.html)
- Create a Label object (see https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/Label.html)
- Add your Button and Label to your BorderPane. See the documentation to determine how to add like the images below: https://docs.oracle.com/javase/8/javafx/api/javafx/scene/layout/BorderPane.html
- Add a click handler to your button, with the following line:
    - `myButton.setOnAction(e -> myLabel.setText("Hello 3035"));`
    - Notice that the above uses a lambda expression for an action handler. Read here for more detail: `https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html#use-case`

- If you have a project template using e(fx)clipse, you can delete the stylesheet and references to it in your class (since you will be doing the styling in code).
- Once you have your program running and behaving correctly, add Javadoc comments to your main class containing your Assignment description, your name, and student number.
- Export your project as a Jar file as described below.

For JavaFX FXML:

- See the HelloWorld examples posted on D2L that were discussed in class, and amend to replicate the HelloWorld you have already done for this assignment.
- Create a new JavaFX project like you did above, then add and JavaFX FXML file (right click on the project again, and access the JavaFX group, select JavaFX FXML Document).
- Load a BorderPane as your root node from the FXML file from your Java code.
- To get started on the content of your FXML file, see: https://docs.oracle.com/javafx/2/fxml_get_started/why_use_fxml.htm#CHDGAFHF
- Add event an event handler, so that your button can change the label value (there are lots of ways to do this, but we will look at the simplest).
- First add an id to your Label element, so that your event handler will be able to reference it by name:

```
<Label text="default text" fx:id="label1" />
```

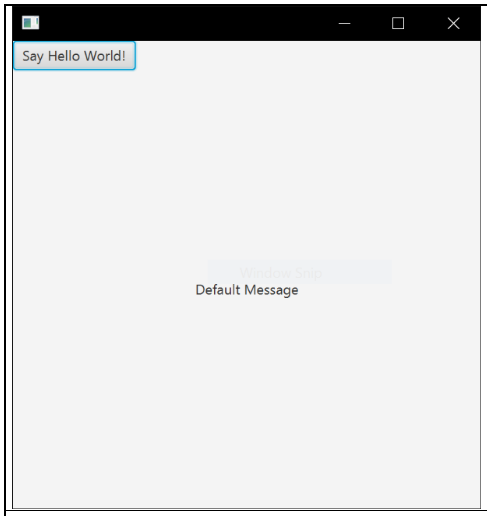
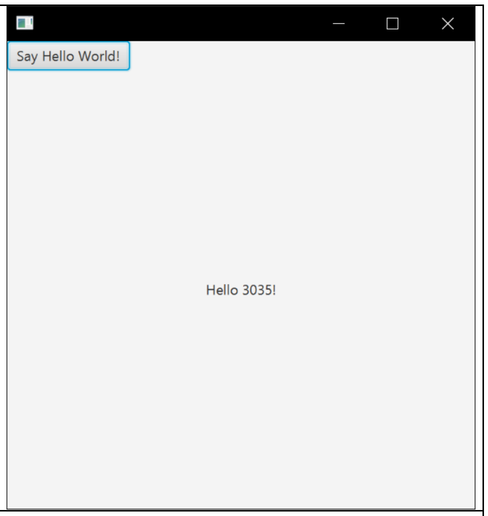
- Next, you will use a little bit of JavaScript (alternatively, you can use any JVM scripting language; Groovy, Koitlin, Clojure) to make the do something, namely, change the text value of the label. Add the following inside your root node (just below the tag for your BorderPane).

```
<fx:script>
    function reactToClick() {
       label1.setText("Button clicked");
     }
</fx:script>
```

- To use JavaScript in your document… don't forget to include a JavaScript declaration right after the XML doctype declaration:

```
<?language javascript?>
```

- See more on event handlers here: https://docs.oracle.com/javase/8/javafx/api/javafx/fxml/doc-files/introduction_to_fxml.html#event_handler_attributes
- Export your project as a Jar file as described below.

	
JavaFX before clicking	JavaFX after clicking

(Note that your UI may look slightly different, which is fine, as long as the label and button are on the screen and in the same position).

What to hand in

- See instructions on exporting your project as a jar file in Eclipse
- A jar file for your JavaFXHelloWorld (make sure that it includes your source files)
- A jar file for your JavaFXMLHelloWorld (make sure that it includes your source files)
- A readme.txt file that includes your name, student number, and indicates exactly what the marker needs to do to run your code and use your system (The instructions are obvious for this assignment, but will be very useful for future assignments. Note that your systems for 3035 should never require the marker to install external libraries or any new downloads).

Where to hand in

Hand in your three files (two jar files and one readme.txt) to the link on the course D2L.

Evaluation

Marks will be given for producing a system that meets the requirements above, and compiles and runs without errors. You will either receive full grades for this assignment or no grades for this assignment. All requirements must be met to receive full grades.

Exporting Assignments as Executable Jar files with Source

Step 1

- Before starting you will need to make sure your project is compiling and running without errors
- Select your Project in the "Package Explorer"
- Right-click your project and select "Export"
- Select "JAR File" from the list (do not select "Runnable JAR file")
- Select your project (make sure only one project is selected)
- Click "Next"

Step 2

- Check the following options -- "Export generated class file" -- "Export Java source files and resources"
- Select a path and name your jar file with an appropriate name.
- Check the following further options -- "Compress the contents of your Jar file" -- "Add directory entries"
- Click "Next"

Step 3

- Click "Next" again... these should probably be all deselected (e.g., export class files with errors)
- "Generate the manifest file" -- You do not need to save the manifest in the workspace
- Do not seal any contents
- Select the main class (this will allow the jar to be executed)
- Click "Finish"

Step 4

- Verify your jar file... double click it. (If it a dialog opens on Windows: choose "open with other application" then navigate to the javaw.exe in your Java installation directory (probably C:\Program Files\Java\Java-version\bin.
- Your file should then execute and run as expected.
- Next verify that your files are stored correctly. Unzip your jar file (your zip program may require you to rename the file to have a .zip extension)
- Verify that you have both the .class and .java (source) files