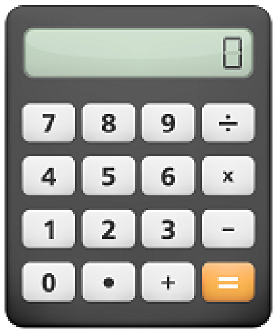
Java FX Part #1

Due Jan 23 by 11:59pm **Points** 100 **Submitting** a website url
Available until Jan 26 at 11:59pm

This assignment was locked Jan 26 at 11:59pm.

Java FX Part #1



This will be the first part of a two-part assignment as an introduction to Java FX. We will be building a simple calculator with a user interface. Our calculator will only operate on integer values. The goal is to work with Java FX layouts, controls and events. This first assignment will focus on layouts and controls. Part #2 of this assignment can be found here: **Java FX Part #2**.

Objectives

Course:

- Implement a program using a modern software framework that leverages architectural patterns, such as model-view-controller, object-relational-mapping or dependency injection.

Module:

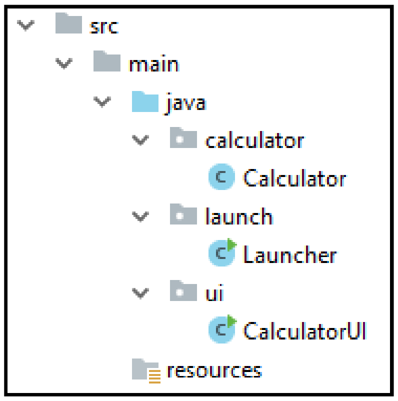
- To write a graphical program using a modern Java framework.
- To identify existing design patterns in a modern framework.
- To assemble an application while avoiding writing redundant code segments.

Setting up Your Project

Create a new Maven Project in IntelliJ with the following settings:

- Group ID: edu.greenriver.it
- Artifact ID: CalculatorApp
- Build settings: Java 8

Your project should have several packages that can be used to separate the various components that are part of your solution. Use the following as a guideline:



With part #1, you are only responsible for implementing your UI class (CalculatorUI) and the main entry point of the application (Launcher). Details for all classes are listed below:

CalculatorUI

- Extends from Application
- Holds all code to build your user interface, as seen below

Launcher

- Has a main() method to launch your application
- Launching a Java FX GUI outside of the Application class is not completely straightforward. Use the following code segment to launch the program

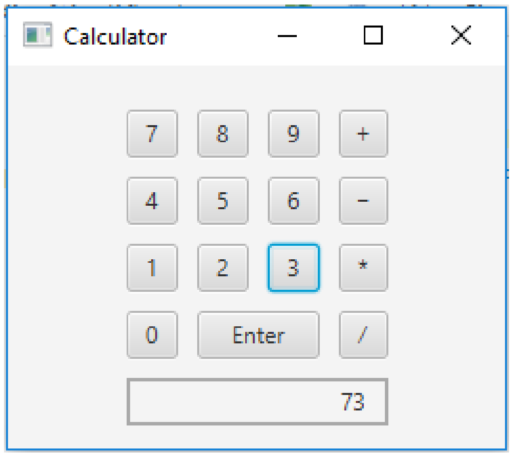
```
Application.launch(CalculatorUI.class, args);
```

Calculator

- Ignore this class for now. It will be completed in part #2.

Building the UI

Your job is to replicate the user interface in the following screenshot.



This UI uses a grid pane to place elements using specific columns and rows. Your design should directly follow the screenshot, including:

- Placement of controls

- Padding around and within elements
- Any borders present
- Text on GUI elements
- Window title

The rectangle that is used for calculator output is actually a Label control. The text in your label should be right aligned. Right text alignment is not supported in Java FX controls currently. Instead you should use an HBox to place your label on the right. To do this use the HBox.setAlignment() method.

The HBox above contains a border using the HBox .setBorder() method. This accepts a javafx.scene.layout.Border object, which is covered here: **https://docs.oracle.com/javase/8/javafx/api/javafx/scene/layout/Border.html (https://docs.oracle.com/javase/8/javafx/api/javafx/scene/layout/Border.html)** .

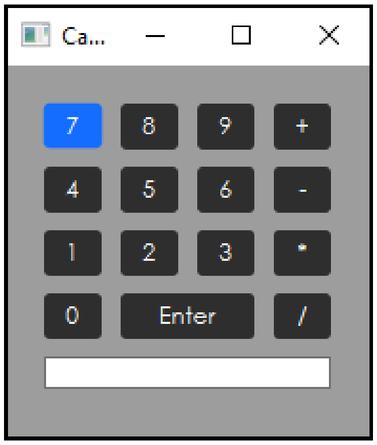
Note: Learning how to use borders programmatically will take some investigation.

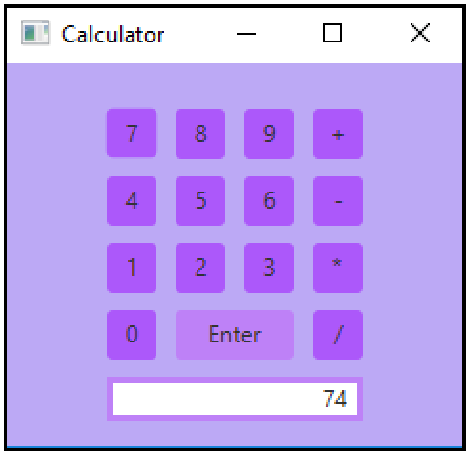
Submission

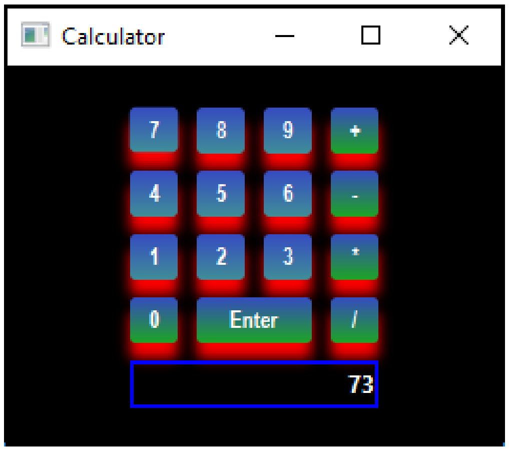
- All projects must be completed through the IntelliJ IDE
- Your project should be a Maven project with appropriate settings
- Your project files should be saved to a private GIT repository which is shared with myself
- All files must have proper documentation (comments and full Javadocs)
- All files must follow our style guidelines

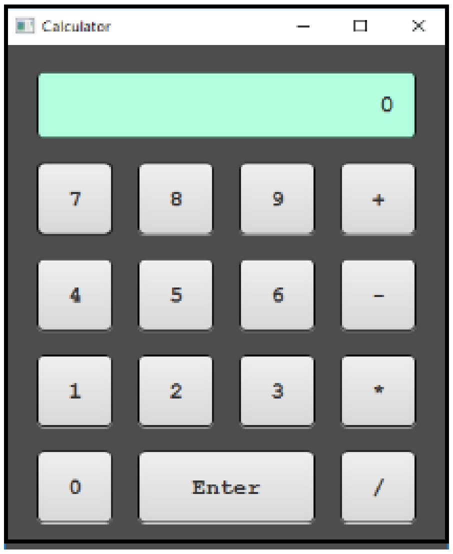
Extra Credit (5 points)

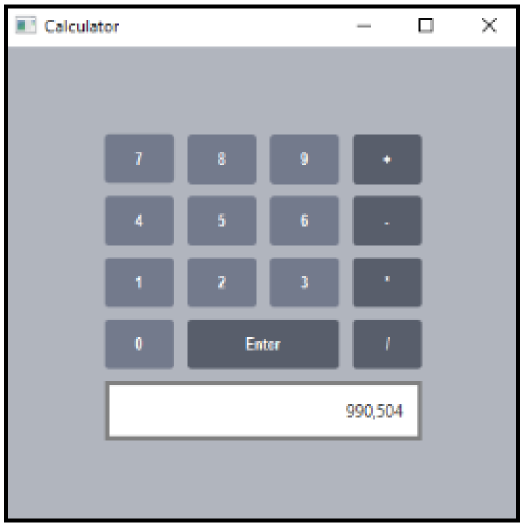
Java FX supports it's own styling language using CSS and custom styles. You can read more about this here: **http://docs.oracle.com/javafx/2/get_started/css.htm#BABBGJBI (http://docs.oracle.com/javafx/2/get_started/css.htm#BABBGJBI)** . Apply font and color changes to your application to "spice" up its look. Points will be awarded based on effort.

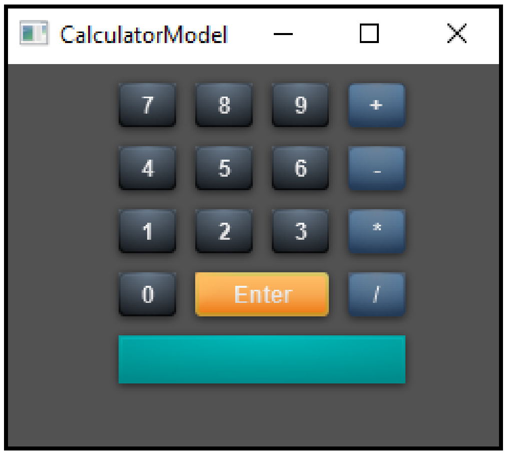












Java FX Assn. Part #1 Rubric

Criteria	Ratings		Pts
Buttons are added to the interface for entering numbers into the calculator, according to the screenshot above. No redundant code is written to display the buttons.	15.0 pts Full Marks	0.0 pts No Marks	15.0 pts
Buttons are added to the interface for basic arithmetic operators: add, subtract, multiply and divide. No redundant code is written to display the buttons.	15.0 pts Full Marks	0.0 pts No Marks	15.0 pts
A label with borders is added to the bottom of the calculator. The label text is right aligned within the border.	15.0 pts Full Marks	0.0 pts No Marks	15.0 pts
All controls are added to a GridPane using the columns and rows seen in the screenshot. The "Enter" button should be placed across two columns.	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
Appropriate padding and spacing are present as seen in the screenshot above. Padding is set for all button controls and labels. The GridPane is center aligned on the application window. Spacing is provided between all controls.	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
The application window is given an appropriate width and height. The title of the window is set to "Calculator."	5.0 pts Full Marks	0.0 pts No Marks	5.0 pts
IntelliJ project is set up correctly as a Maven project. All project files are saved to GIT in a private repository.	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
Full Javadocs are provided for all classes and public methods.	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
All code follows the style guide provided.	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
			Total Points: 100.0