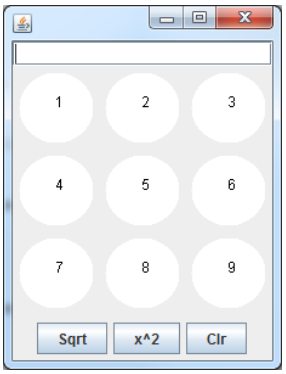
GEEN165 Lab – Event Handling

Description:



In this lab exercise, you will create a simple JavaFX calculator. The GUI will have a panel of graphic buttons (filled ovals) with string labels. When pressed, the graphic button background will change color indicating the button is currently pressed. When the mouse is released, the background will return to its original color. Also, the string value in the button will be concatenated to the TextField at the top of the GUI. There will be additional command Buttons at the bottom of the GUI with functionality described below.

CircleButton Class:

The graphic circular buttons are created by drawing a filled circle on a StackPane. So, the pictured GUI uses 9 different StackPanes for displaying the 9 graphic buttons. Of course, these CircleButton objects can then be placed on a single GridPane to achieve the 3x3 layout (see SimpleCalc class below). Create a class named CircleButton that extends the StackPane class. The class should have a Label property and a Circle property. By default, the background of the circle should be white and the text black. The no-arg constructor sets the colors to their defaults and the text to the empty string "". A second constructor should have a String parameter to initialize the Label property. Set the preferred size of the CircleButton pane to 100 but the radius of the circle to 90 pixels. Put this class in a separate file.

CircleButton	
-lblValue : Label -circle : Circle	Displays button numeric value Circle object that is displayed.
+CircleButton() +CircleButton(val : String) +setColor(Paint color) : void +getColor() : Paint +getValue() : String	Create a button with no displayed value and a white background. Create a button with val as the numeric label and a white background. Set the background/fill color of the CircleButton. Get the current fill color of the CircleButton. Get the value currently displayed on the CircleButton.

SimpleCalc Class (extends Application):

This is your main project class that contains your *start()* method. This class will create a BorderPane with a TextField at the top, a GridPane of 9 CircleButtons in the center and an HBox of three JButtons at the bottom. When adding the CircleButtons, you should use nested for-loops to facilitate the instantiation of the CircleButtons with the correct text label (1..9 or 0..8) and the registration of a single Listener for all nine CircleButtons.

Mouse Events:

Create an inner class that implements the EventHandler<MouseEvent> interface. Instantiate an object of the inner class and register it with all nine CircleButtons. The handle() method should toggle the clicked CircleButton background to a new color using the *setFill()* method. In addition, the numeric value on the CircleButton should be appended to the current value in the top TextField. On mouse release, change the background back to the original color. You must call the methods for registering mouse pressed and mouse released for each CircleButton.

Algorithm for the handle() method registered with the CirleButtons:

- Use the event object passed to handle() to get a reference to the CircleButton that was clicked.
- Check the current fill color of the CircleButton
    - If the fill color is white, this is a mouse press event so change the fill color (yellow?) and append the value on the button to the current TextField value (mouse pressed).
    - Else change the fill color back to white (mouse released).

Command Buttons:

The Buttons in the bottom HBox pane will perform the following functions on the current value in the TextField:

- x^2 – Replace the current TextField with the square of the current value in the TextField
- sqrt - Replace the current TextField with the square root of the current value in the TextField
- Clear - Clear the contents of the TextField

You can create another inner class that implements the EventHandler interface to handle the button click events or you can create anonymous listeners for the Buttons.

Grading:

10pts – GUI with no event handling.

15pts – GUI plus circle button functionality. A mouse press causes CircleButton background to change. The mouse release causes the circle button value to be appended onto the TextField.

20pts – Command Button functions implemented.