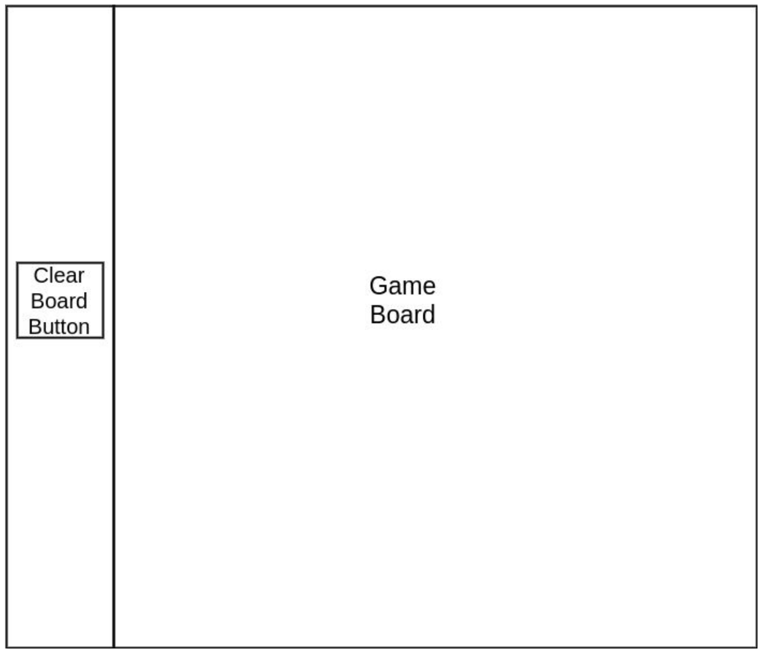
CSc 210: Software Development
Section 11: GUI and Event Listeners
November 13th, 2017

Today we will explore designing a GUI and adding in event listeners. We have began to discuss in class about what event driven programming is all about. Now, we will actually create an interface for a tic tac toe game and add in listeners to different aspects.

Part I: Setting up the GUI

1. First, open a file in your editor of choice called Drawing.java and create a class called Drawing that extends Application. Create a main method that calls launch and also override launch as we have seen in class. Add in the appropriate imports.

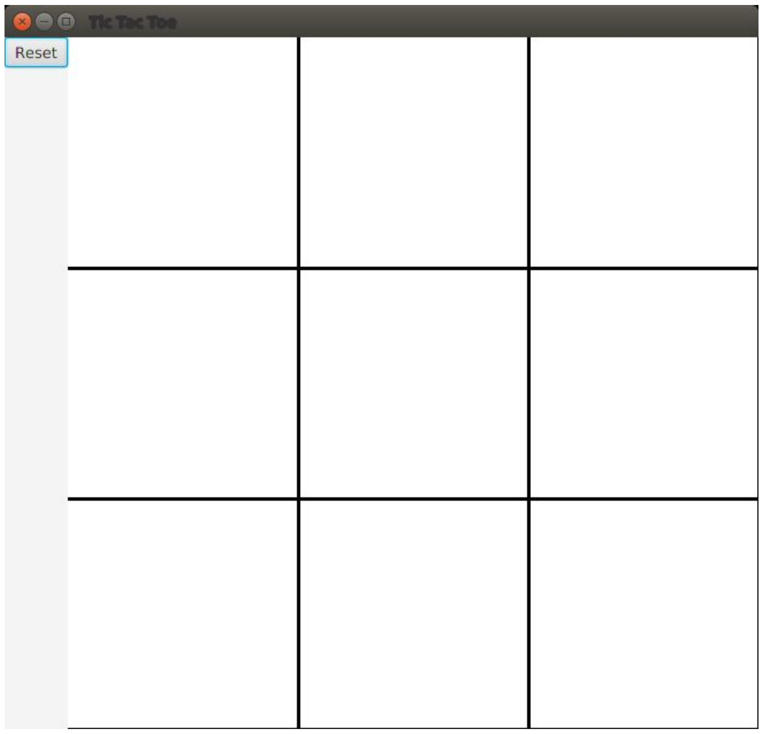
2. Now, we will start laying out the GUI. We want it to look like the following:



Consider the different types of panes discussed in class. What would work here? Essentially, you want to split the screen vertically, having one side contain a button and the other contain the tic tac toe game board.

3. Once you have figured out how to lay out the GUI, create a button object and add it into the left side. Additionally, create a canvas object (its dimension should be square) and it it to the right side.

4. Now, we want to set up the lines that divide the tic tac toe board up. To do this, first pull the GraphicsContext out of the canvas. Then, create a method that evenly divides the board up into squares as shown below.



5. We now have a functional GUI to create our tic tac toe game.

Part II: Event Listeners

1. To play tic tac toe, we want to actually be able to place X's and O's on the board. We will create an event listener to detect when the mouse is clicked. To do this, we will create an inner class. An inner class is simply a class defined within another, outer class. Note that inner classes have access to any member variables of the outer class. Create a private class called MyMouseEvent that implements EventHandler<MouseEvent> inside of the Drawing class.

2. Inside of MyMouseEvent, you will have to override the method **public void handle(MouseEvent e)**. In this method is where you will place any code you want to be executed whenever the mouse is clicked on the object the handler is

listening to. We will be adding this to our Canvas object in our GUI to detect when it is clicked. When the mouse is clicked, we want to place an X inside of whichever tic tac toe box was clicked. If this is the functionality we want, what code should go inside of this method?

3. Once you have implemented this private class, we must create an instance of it and add it as a listener to our canvas. Go to where you create your Canvas for your GUI. Create a new MyMouseEvent object and then call the Canvas method addEventHandler() to add it as a listener. This method takes two parameters, first, the type of event you are listening for (this will be MouseEvent.MOUSE_CLICKED) and second, the EventHandler object to handle this event (this will be the instance of your private class you just created).

4. Test the functionality of your listener and if you have questions, ask your SL.

5. Now, we will also implement a listener for your reset button. Create another private class similar to as we did before and name it whatever you would like. There are two differences, you will now implement EventHandler<ActionEvent> and you will override the method **public void handle(ActionEvent e)**. The functionality of this method should be to clear any X's and O's from the canvas when it is clicked.

6. Now, go to where you create our button and create a new instance of your button handler. Then, add it as a listener on your button by calling the method setOnAction from your button and passing the button listener as a parameter.

7. Test the functionality of your reset button. If you have questions, ask your SL.

8. Finally, you should add in alternating moves. The first click should produce an X, the second click on O, so on and so forth.

9. Once you get that functionality working, you should be done!