Coral Marquez
10/21/15 (07)

Method Madness Essay

My project, “Marquez_7_MethodMadness,” creates a staircase with different colors. My project is unique in the way that it is very colorful, and bright. The way the project is designed is to start at the back, and slowly decreases in length while also travelling towards the right (The bottom shape is at the top left corner while “travelling down the stairs” to the center).

The for loop is designed, so that every *}else if (){* statement, a different method is being called on to draw a different rectangle. The integers a, b, and c are used to change the shape and location of the rectangles. Since the rectangles are not given variables in the in the methods, it allows me to set up a system to where each time a number goes through the *}else if (){* statement a number is added or subtracted from the value given in the previous statement. This is how the staircase is possible, because the values change at a constant rate.

Coral Marquez
10/21/15 (07)

```
    public void start(Stage primaryStage) {
21      primaryStage.setTitle("Drawing Operations Test");
22      Group root = new Group();
23      Canvas canvas = new Canvas(1000, 1000);
24      GraphicsContext gc = canvas.getGraphicsContext2D();
25      int a = 0;
26      int b = 205;
27      int c = 500;
28      for (int i = 0; i < 1000; i++) {
29          if (i == 0) {
30              drawShape0(gc, a, b, c);
31          } else if (i == 1) {
32              drawShape1(gc, a, b, c);
33          } else if (i == 2) {
34              drawShape2(gc, a, b, c);
35          } else if (i == 3) {
36              drawShape3(gc, a, b, c);
37          } else if (i == 4) {
38              drawShape4(gc, a, b, c);
39          } else if (i == 5) {
40              drawShape5(gc, a, b, c);
41          } else if (i == 6) {
42              drawShape6(gc, a, b, c);
43          } else if (i == 7) {
44              drawShape7(gc, a, b, c);
45          } else if (i == 8) {
46              drawShape8(gc, a, b, c);
47          } else if (i == 9) {
48              drawShape9(gc, a, b, c);
49          }
50          a = a + 50;
51          b = b + 20;
52          c = c - 50;
```

My project uses forms of encapsulation to keep things organized, and to keep a “cap” on the methods not being used at a certain point of a project. Since in this project methods were used a lot, keeping them organized was very important.

Here are the different methods I used:

Coral Marquez
10/21/15 (07)

```
60      private void drawShape0(GraphicsContext gc, int x, int y, int z) {
61          gc.setFill(Color.AQUA);
62          gc.fillRect(0, x, y, z);
63      }
64
65      private void drawShape1(GraphicsContext gc, int x, int y, int z) {
66          gc.setFill(Color.CORNFLOWERBLUE);
67          gc.fillRect(50, x, y, z);
68      }
69
70      private void drawShape2(GraphicsContext gc, int x, int y, int z) {
71          gc.setFill(Color.AQUAMARINE);
72          gc.fillRect(100, x, y, z);
73      }
74
75      private void drawShape3(GraphicsContext gc, int x, int y, int z) {
76          gc.setFill(Color.YELLOW);
77          gc.fillRect(150, x, y, z);
78      }
79
80      private void drawShape4(GraphicsContext gc, int x, int y, int z) {
81          gc.setFill(Color.FUCHSIA);
82          gc.fillRect(200, x, y, z);
83      }
84
85      private void drawShape5(GraphicsContext gc, int x, int y, int z) {
86          gc.setFill(Color.HOTPINK);
87          gc.fillRect(250, x, y, z);
88      }
89
90      private void drawShape6(GraphicsContext gc, int x, int y, int z) {
91          gc.setFill(Color.AQUA);
92          gc.fillRect(300, x, y, z);
93      }
94
95      private void drawShape7(GraphicsContext gc, int x, int y, int z) {
96          gc.setFill(Color.CORNFLOWERBLUE);
97          gc.fillRect(350, x, y, z);
98      }
99
100     private void drawShape8(GraphicsContext gc, int x, int y, int z) {
101         gc.setFill(Color.AQUAMARINE);
102         gc.fillRect(400, x, y, z);
103     }
104
105     private void drawShape9(GraphicsContext gc, int x, int y, int z) {
106         gc.setFill(Color.YELLOW);
107         gc.fillRect(450, x, y, z);
108     }
109
110 }
111
```

Coral Marquez
10/21/15 (07)

The method *gc.setFill(Color.AQUAMARINE);* gives a color, not a numerical value. Since my numerical values are given by the for loop I used my methods to set a shape and color for my animations. The line *gc.fillRect(100, x, y, z);* gives my drawing a shape, but the dimensions and coordinates for the rectangle are not yet defined until the for loop. This allows my shapes to change without having to insert individual variables for each value needed.

```
69
70      private void drawShape2(GraphicsContext gc, int x, int y, int z) {
71          gc.setFill(Color.AQUAMARINE);
72          gc.fillRect(100, x, y, z);
73      }
```

All the colors come from *import javafx.scene.paint.Color;*

```
8   import javafx.scene.paint.Color;
```

My canvas (where all the shapes are drawn) is 835 x 500, my canvas begins in the top left corner where the increasing variables travel down to the right. The canvas is imported by the line *import javafx.scene.canvas.Canvas;*.

```
22          Canvas canvas = new Canvas(835, 500);
```

```
6   import javafx.scene.canvas.Canvas;
```

The errors that occurred in my project was mostly based on where to place my methods, and where to call the methods. I had enormous difficulty in this area. I also had difficulty on how to change the coordinate of the rectangles without having to type in a coordinate each time, but after playing around, and help from my colleagues I accomplished what I wanted.

While there are more than 5 methods in "Marquez_7_MethodMadness," only one method is in the *main(): launch(args);*. The reason for this is because the methods are drawn on a canvas and not printed words or numbers.

Coral Marquez
10/21/15 (07)

```
11  public class Marquez_7_MethodMadness extends Application {
12
13      public static void main(String[] args) {
14          launch(args);
15      }
```

In this project I was able to accomplish what I set out to do. I was able to have a successful staircase that descended at a constant rate in a for loop. Although this project was very difficult, and time consuming. I learned a lot, and now I feel I have a greater hold on javaFX.