Jason Mejia 10/15/2019
Professor Auda
CSC221

Project #1 Report

Objective:

Design a class hierarchy of shapes with a super class named MyShape and the subclasses, MyLine, MyCircle, and MyPolygon. MyShape is an abstract class with abstract functions draw and toString. In the MyShape also has the get and set methods for x, y, and for a Color.

Solution Method:

For the solution I have attempted, I first developed the class I needed for the rest which is the MyShape class. The class is abstract meaning it is only meant to be inherited and cannot be used to directly create instance variables. Its methods, draw and toString must be overridden for each of the subclasses. Then I started on the subclasses

What we used for the project is JavaFX, which is a GUI used in Java. One of the functions I used is canvas to create the shapes. In the main program, I created a class that would draw the shapes in the layout I wanted. One function draws the line going diagonal across the box using the MyLine class and the other classes are creating the shapes of rectangles and ovals. To create different colors, I did not create the colors as requested. But I populated a color array and iterated through the array for different colors. I included a test of all the toString methods in my output below.

Project Code:

MyShape.java

```
package Shape;

import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;

public abstract class MyShape {

   protected double x, y;
   protected Color color;

   public double getX() {
       return x;
   }
   public double getY(){
```

```
        return y;
    }
    public Color getColor() {
        return color;
    }
    public void setX(float x) {
        this.x = x;
    }
    public void setY(float y) {
        this.y = y;
    }
    public void setColor(Color color) {
        this.color = color;
    }
    /*
    Methods to be overridden:
    */
    public abstract void draw(GraphicsContext gc);
    public abstract String toString();

}
```

MyLine.java

```
package Shape;

import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;

public class MyLine extends MyShape {
    public MyLine(double x1, double y1, double x2, double y2, Color color) {
        this.x = x1;
        this.y = y1;
        this.x2 = x2;
        this.y2 = y2;
        this.color = color;
    }

    @Override
    public void draw(GraphicsContext gc) {
        gc.setStroke(color);
        gc.strokeLine(x, y, x2, y2);
    }
    @Override
    public String toString() {
        double length = Math.sqrt(((x2 - x) * (x2 - x)) * ((y2 - y) * (y2 - y)));
        double theta = (Math.atan2((y2 - y), (x2 - x)))/(Math.PI/180);
        return "Length: " + length + "px, Angle: " + theta;
    }
    private double x2, y2;
}
```

MyOval.java

```
package Shape;

import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;

public class MyOval extends MyShape {

   protected double width, height, radius;

   public MyOval(){}
   public MyOval(double originX, double originY, double width, double height, Color
color) {
       this.x = originX - (width / 2);     //(x, y) is actually the upper left bound
so it is converted
       this.y = originY - (height / 2);
       this.height = height;
       this.width = width;
       this.color = color;
       this.radius = Math.min(width, height);
   }
   @Override
   public void draw(GraphicsContext gc) {
       gc.setFill(color);
       gc.fillOval(x, y, width, height);
   }

   @Override
   public String toString() {
       return "Center: (" + x + ", " + y + ") " +
               "Width: " + width +
               "Height: " + height;
   }
}
```

MyRectangle.java

```
package Shape;

import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;

public class MyRectangle extends MyShape {
   private double width, height;
   public MyRectangle(double originX, double originY, double width, double height,
Color color) {
       this.x = originX - (width / 2);
       this.y = originY - (height / 2);
       this.width = width;
       this.height = height;
       this.color = color;
   }
   @Override
   public void draw(GraphicsContext gc) {
```

```
        gc.setFill(color);
        gc.fillRect(x, y, width, height);
    }

    @Override
    public String toString() {
        return "Center: (" + x + ", " + y + ") " +
                "Width: " + width +
                "Height: " + height +
                "Area: " + width * height;
    }
}
```

DrawDesign.java

```
import Shape.MyLine;
import Shape.MyOval;
import Shape.MyRectangle;
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.stage.Stage;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;

public class DrawDesign extends Application {

      private Color[] colorsArray = {Color.BLUE, Color.GREEN, Color.YELLOW,
Color.BLUEVIOLET,
                  Color.RED, Color.AQUA, Color.CRIMSON, Color.CHOCOLATE};

      public void xWithLines(GraphicsContext gc) {
            double height = gc.getCanvas().getHeight();
            double length = gc.getCanvas().getWidth();

            MyLine line1 = new MyLine(0, 0, 0, height, Color.BLACK);
            MyLine line2 = new MyLine(0, 0, length, 0, Color.BLACK);
            MyLine line3 = new MyLine(0, 0, length, height, Color.BLACK);

            line1.draw(gc);
            line2.draw(gc);
            line3.draw(gc);
        }

       public void rectangleOvals(GraphicsContext gc) {
            double canvasLength = gc.getCanvas().getWidth();
            double canvasHeight = gc.getCanvas().getHeight();
            double centerX = canvasLength / 2;
            double centerY = canvasHeight / 2;
            double height = canvasHeight * .66;
            double width = canvasLength * .66;
            int colorPick = 0, colorPick2 = 7;

            for(int i = 0; i < 3; i ++) {
```

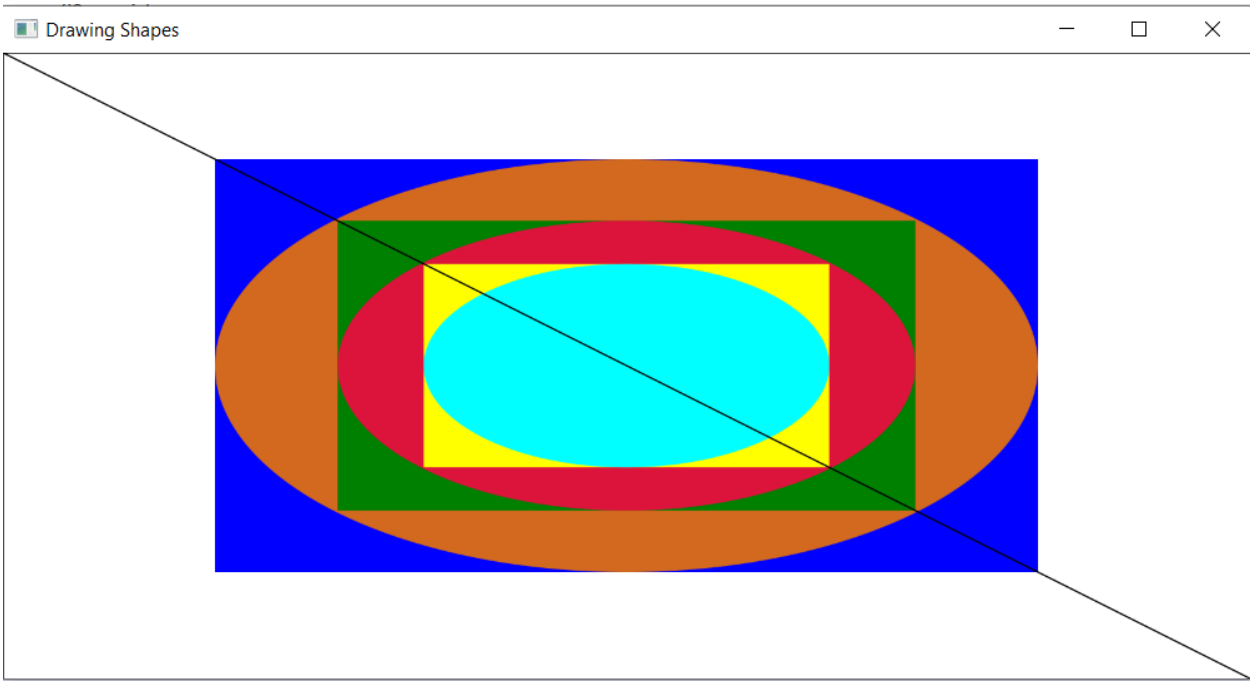
```
                MyRectangle rect = new MyRectangle(centerX, centerY, width, height,
colorsArray[colorPick]);
                rect.draw(gc);
                MyOval oval = new MyOval(centerX, centerY, width, height,
colorsArray[colorPick2]);
                oval.draw(gc);
                height = height * .702;
                width = width * .702;
                colorPick ++;
                colorPick2 --;
            }
        }


        @Override
        public void start(Stage stage)
        {
            Canvas canvas = new Canvas(800, 400);
            GraphicsContext gc = canvas.getGraphicsContext2D();

            DrawDesign x = new DrawDesign();
            x.rectangleOvals(gc);
            x.xWithLines(gc);


           MyRectangle a = new MyRectangle(500, 500, 100, 100, Color.BLACK);
            MyLine b = new MyLine(0, 0, 45, 45, Color.RED);
            MyOval c = new MyOval(50, 50, 10, 10, Color.AQUA);
            System.out.println(a.toString());
            System.out.println(b.toString());
            System.out.println(c.toString());

            Pane root = new Pane();
            root.getChildren().add(canvas);
            Scene scene = new Scene(root);
            stage.setScene(scene);
            stage.setTitle("Drawing Shapes");
            stage.show();
        }
    }
```



Console

DrawDesign [Java Application] C:\Program Files\Java\jdk-13\bin\javaw.exe (Oct 15, 2019, 1:49:48 AM)

```
Center: (450.0, 450.0) Width: 100.0Height: 100.0Area: 10000.0
Length: 2025.0px, Angle: 45.0
Center: (45.0, 45.0) Width: 10.0Height: 10.0
```