Zi Xuan Li

Professor Auda

CSC 22100

March 7th, 2023

Assignment #1

Statement of the problem:

This purpose of this assignment is to build a JavaFX **BorderPane** layout that will be used to display outputs in the upcoming assignments.

Solution methods:

The **BorderPane** layout will contain two regions: a left region and a center region. The left region will be a **VBox** layout that contains a **Label** object displaying the text "MyColor Palette" and an object titled **MyColorPalette** showing a set of colors from the enum **MyColor** for color selection. The center region will be a **Canvas** layout painted in a color from **MyColor** selected from **MyColorPalette** and displays **MyColor** name.

Enum **MyColor** will be used by class **MyColorPalette** to define the red, green, blue, and opacity components of each color with a value in the range of [0 – 255].

Class **MyColorPalette** will use a **TilePane** layout comprised of 144 square **Pane** layouts each displaying a different color from **MyColor**, such that, when a tile is clicked the canvas in the center region is painted in the color selected.

Class **TestMyColor** will contain the code for the **VBox**, **Canvas**, as well as the code to launch the **Stage** which will resemble the JavaFX **BorderPane** layout that is demonstrated in the assignment guidelines.

All classes that imported:

- javafx.application.Application: This class is the starting point for JavaFX applications. It provides the main method for launching the JavaFX runtime and starting the application.
- javafx.geometry.Insets: This class represents an immutable set of insets that can be used to specify the spacing between nodes in a layout.
- javafx.scene.Scene: This class represents the scene graph for a JavaFX application. It provides a container for all the visual elements that make up the user interface of an application.
- javafx.scene.canvas.Canvas: This class represents an area within which you can draw graphics using a GraphicsContext.
- javafx.scene.canvas.GraphicsContext: This class provides the main rendering API for the Canvas class. It allows you to draw shapes, images, and text on a Canvas.
- javafx.scene.control.Label: This class represents a text label that can be used to display text in a JavaFX application.
- javafx.scene.layout.*: This package contains several classes for laying out nodes in a JavaFX application. In this case, the wildcard import imports all the classes in this package.
- javafx.scene.paint.Color: This class represents a color in the JavaFX scene graph. It provides methods for creating colors, manipulating colors, and converting colors to different color models.

- javafx.scene.text.Font: This class represents a font that can be used to render text in a JavaFX application.
- javafx.stage.Stage: This class represents the primary stage of a JavaFX application. It provides methods for controlling the size, position, and visibility of the stage.
- javafx.scene.Node: This class represents a node in the JavaFX scene graph. It is the base class for all the visual elements that make up the user interface of an application.
- java.util.Optional: This class represents an optional value that may or may not be present. It is often used as a return type for methods that may or may not return a value.
- javafx.geometry.Orientation: This class represents the orientation of a control or layout node. It can be either horizontal or vertical.
- java.util.Arrays: This class contains various methods for working with arrays in Java. In this case, it is being used to convert an array of MyColor objects to an array of strings.

```java
import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.Label;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Stage;
import javafx.scene.Node;
import java.util.Optional;
import javafx.geometry.Orientation;
import java.util.Arrays;
```

Java code:

TestMyColor.java: This java code defines a JavaFX application called *TestMyColor* that creates a color palette and a canvas that displays the selected color.

- The *MyColor* class is referenced to define color values. These values are used to create a color palette.
- The *addLeftVBox* method is used to create a VBox that contains the color palette on the left-hand side of the application window.
- The *addCenterCanvas* method creates a canvas that displays the selected color in the center of the application window.
- The *start* method initializes the application window and sets up the layout. It creates a *BorderPane* and sets the *VBox* containing the color palette on the left side of the pane, and the canvas displaying the selected color to the center of the pane.
- When the user clicks on a color tile in the color palette, the *setOnMouseClicked* method updates the canvas to display the selected color.

```java
package com.example.assignment1;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.Label;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Stage;
import javafx.scene.Node;
import java.util.Optional;

public class TestMyColor extends Application {
    MyColor [] myColors = MyColor.getMyColors();
    int sizeMyColor = myColors.length;

    public VBox addLeftVBox(double widthLeftCanvas, double heightCanvas,
TilePane TP, MyColor color) {

        VBox VB = new VBox();
        VB.setPrefWidth(widthLeftCanvas);
        VB.setPadding(new Insets(5));

        Label lblMyColorPalette = new Label("MyColor Palette");
        lblMyColorPalette.setPrefWidth(widthLeftCanvas);
        lblMyColorPalette.setTextFill(MyColor.WHITE.getJavaFXColor());
        lblMyColorPalette.setBackground(new Background(new
BackgroundFill(Optional.ofNullable(color).orElse(MyColor.GREY).getJavaFXColor
(), CornerRadii.EMPTY, Insets.EMPTY)));

        VB.getChildren().addAll(lblMyColorPalette, TP);

        return VB;
    }

    public Canvas addCenterCanvas (double widthCanvas, double heightCanvas,
MyColor color) {

        MyColor colorPicked =
Optional.ofNullable(color).orElse(MyColor.WHITE);
        Canvas CV = new Canvas(widthCanvas, heightCanvas);
        GraphicsContext GC = CV.getGraphicsContext2D();

        GC.clearRect( 0, 0, widthCanvas, heightCanvas);
        GC.setFill(colorPicked.getJavaFXColor());
        GC.fillRect( 0, 0, widthCanvas, heightCanvas);

        double xText = 5.0; double yText = 20.0;
        GC.setStroke(colorPicked.invertColor());
        GC.setFont(Font.font ("Calibri", 13));
        GC.strokeText(colorPicked.toString(), xText, yText);

        return CV;
    }
    @Override
    public void start(Stage stage) {
        BorderPane BP = new BorderPane();
        Pane leftPane = new Pane(); Pane centerPane = new Pane();

        double widthCanvas = 600; double heightCanvas = 300;
        double widthLeftCanvas = 0.4 * widthCanvas;
        double widthCenterCanvas = widthCanvas - widthLeftCanvas;

        MyColorPalette CP = new MyColorPalette(widthLeftCanvas,
heightCanvas);
        TilePane TP = CP.getPalette();

        leftPane.getChildren().add(addLeftVBox(widthLeftCanvas, heightCanvas,
TP, MyColor.BLACK));
        BP.setLeft(leftPane);

        centerPane.getChildren().add(addCenterCanvas(widthCenterCanvas,
heightCanvas, null));
        BP.setCenter(centerPane);

        TP.setOnMouseClicked(e -> {

            MyColor color = CP.getColorPicked(); String tileID =
color.toString();
            for (Node tile : TP.getChildren()) {
                if (tile.getId() == tileID) {

centerPane.getChildren().add(addCenterCanvas(widthCenterCanvas, heightCanvas,
color));
                    BP.setCenter(centerPane);
                    break;
                }
            }
        });

        stage.setTitle("Assignment #1");
        Scene scene = new Scene(BP, Color.WHITE);
        stage.setScene(scene);
        stage.setHeight(320);
        stage.setResizable(false);
        stage.show();
    }
}
```

MyColorPalette.java: This java code defines a class called *MyColorPalette* which represents a color palette in the JavaFX application.

- The class has a private instance variable *colorPicked* of type *MyColor*, which represents the currently selected color.
- The class has a private instance variable colors of type *MyColor*[], which is an array of all the possible colors in the palette.
- The class has a private instance variable *sizeMyColor* of type int, which represents the size of the colors array.
- The class has two public methods *setColorPicked* and *getColorPicked* which respectively set and return the currently selected color.

- The class has a constructor that takes in two double values *widthPalette* and *heightPalette*, which are used to calculate the size of each color tile in the palette.
- The class has a public method *getPalette* which returns a *TilePane* object that represents the color palette. The *TilePane* object contains a set of Pane objects, each representing a color tile.
- The *getPalette* method loops over the colors array and creates a Pane object for each color tile. It sets the id property of the Pane object to the *toString* representation of the color and sets the background property of the Pane object to a *BackgroundFill* object that uses the JavaFX color corresponding to the *MyColor* object.
- The method also attaches an event handler to each Pane object that listens for mouse clicks. When a tile is clicked, the event handler sets the *colorPicked* property of the *MyColorPalette* object to the *MyColor* object corresponding to the clicked tile.

```java
package com.example.assignment1;

import javafx.geometry.Insets;
import javafx.geometry.Orientation;
import javafx.scene.layout.*;

import java.util.Arrays;

public class MyColorPalette {
    MyColor colorPicked;
    MyColor [] colors = MyColor.getMyColors();
    int sizeMyColor = colors.length;
    double widthTile, heightTile;

    public MyColorPalette(double widthPalette, double heightPalette) {
        this.widthTile = widthPalette / 12.0 - 1.0;
        this.heightTile = this.widthTile;
    }
    public void setColorPicked(MyColor color){
        colorPicked = color;
    }
    public MyColor getColorPicked(){
        return colorPicked;
    }
    public TilePane getPalette() {

        TilePane TP = new TilePane();
        TP.setPrefTileWidth(widthTile); TP.setPrefTileHeight(heightTile);
        TP.setPrefRows(12);
        TP.setOrientation(Orientation.HORIZONTAL);
        TP.setPadding(new Insets( 1));

        for (int j = 0; j < sizeMyColor; j++){
            MyColor color = colors[j];
            String tileId = color.toString();

            Pane tileMyColor = new Pane();
            tileMyColor.setId(tileId);
            tileMyColor.setBackground(new Background(new
BackgroundFill(color.getJavaFXColor(), CornerRadii.EMPTY, Insets.EMPTY)));
```

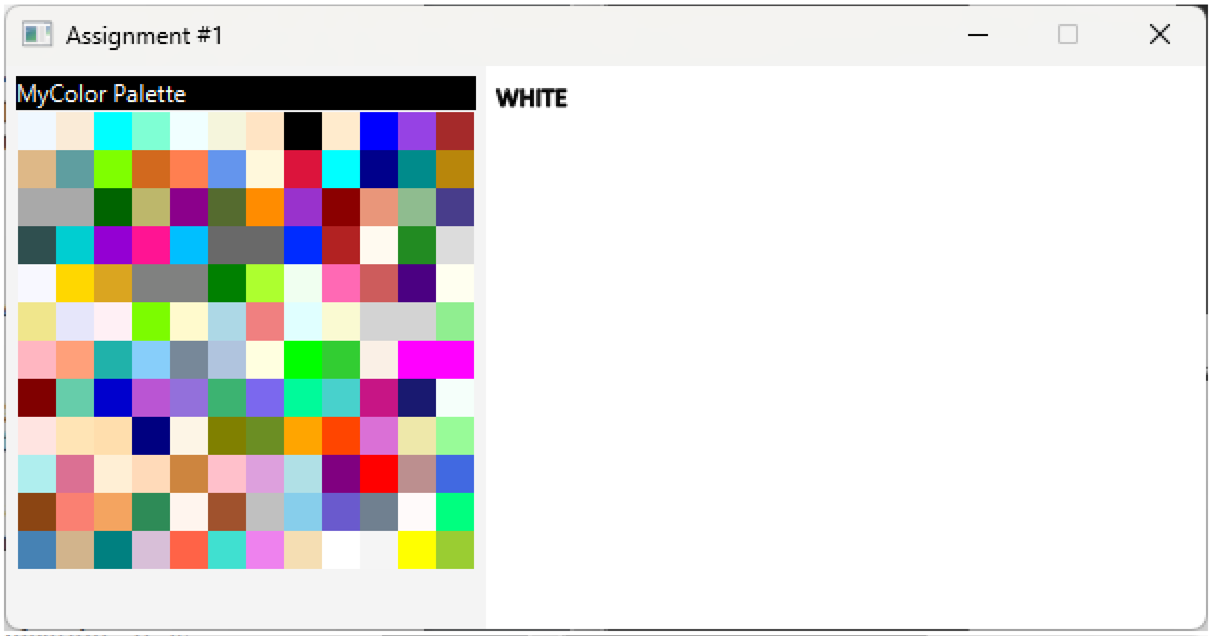
```
            tileMyColor.setOnMouseClicked(e -> {
                MyColor colorClicked =
colors[Arrays.asList(MyColor.getMyColorIds()).indexOf(tileId)];
                setColorPicked(colorClicked);
            });

            TP.getChildren().add(tileMyColor);
        }
        return TP;
    }
}
```

MyColor.java: This java code defines an enumeration type called *MyColor* that contains a list of named constants representing colors.

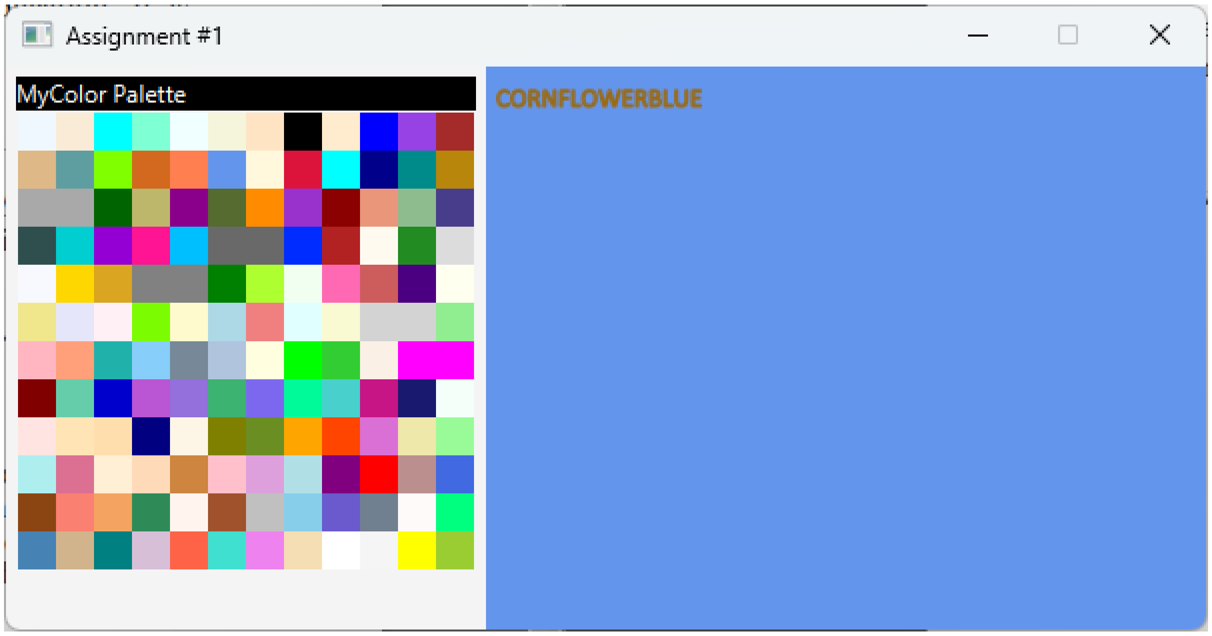
- Each named constant is defined with a specific RGB (red, green, blue) color value represented as 4 integers between 0 and 255, inclusive. The first three integers represent the red, green, and blue component of the color, respectively, while the last integer represents the alpha (opacity) of the color.
- The *MyColor* class has a constructor that takes four integer arguments for the red, green, blue, and alpha components, and sets them using the *setR()*, *setG()*, *setB()*, *setA()* methods respectively.
- The *getR()*, *getG()*, *getB()*, and *getA()* methods return the corresponding color component.
- The *getJavaFXColor()* method returns a JavaFX *Color* object that corresponds to the color represented by *MyColor* object, with the alpha value converted from an integer in the range of 0 – 255 to a floating point value in the range 0.0 – 1.0.
- The *getMyColors()* method returns an array of *MyColor* object that includes all possible color combinations represented by the *MyColor* class.
- The *getMyColorIds()* method returns an array of strings that includes the string representation of all possible color combinations represented by the *MyColor* class.
- The *invertColor()* method returns a JavaFX Color object that represents the inverted color of the *MyColor* object, with each color component subtracted from 255 and the alpha value converted from an integer in the range 0-255 to a floating-point value in the range 0.0-1.0.

```
package com.example.assignment1;

import javafx.scene.paint.Color;
import java.util.Random;

enum MyColor{
    ALICEBLUE(240, 248, 255, 255),
    ANTIQUEWHITE(250, 235, 215, 255),
    AQUA(0, 255, 255, 255),
    AQUAMARINE(127, 255, 212, 255),
    AZURE(240, 255, 255, 255),
    BEIGE(245, 245, 220, 255),
    BISQUE(255, 228, 196, 255),
    BLACK(0, 0, 0, 255),
    BLANCHEDALMOND(255, 235, 205, 255),
    BLUE(0, 0, 255, 255),
    BLUEVIOLET(138, 43, 226, 225),
    BROWN(165, 42, 42, 255),
    BURLYWOOD(222, 184, 135, 255),
    CADETBLUE(95, 158, 160, 255),
    CHARTREUSE(127, 255, 0, 255),
    CHOCOLATE(210, 105, 30, 255),
    CORAL(255, 127, 80, 255),
    CORNFLOWERBLUE(100, 149, 237, 255),
    CORNSILK(255, 248, 220, 255),
    CRIMSON(220, 20, 60, 255),
    CYAN(0, 255, 255, 255),
    DARKBLUE(0, 0, 139, 255),
    DARKCYAN(0, 139, 139, 255),
    DARKGOLDEN(184, 134, 11, 255),
    DARKGRAY(169, 169, 169, 255),
    DARKGREY(169, 169, 169, 255),
    DARKGREEN(0, 100, 0, 255),
    DARKKHAKI(189, 183, 107, 255),
    DARKMAGENTA(139, 0, 139, 255),
    DARKOLIVEGREEN(85, 107, 47, 255),
    DARKORANGE(255, 140, 0, 255),
    DARKORCHID(153, 50, 204, 255),
    DARKRED(139, 0, 0, 255),
    DARKSALMON(233, 150, 122, 255),
    DARKSEAGREEN(143, 188, 143, 255),
    DARKSLATEBLUE(72, 61, 139, 255),
    DARKSLATEGRAY(47, 79, 79, 255),
    DARKTURQUOISE(0, 206, 209, 255),
    DARKVIOLET(148, 0, 211, 255),
    DEEPPINK(255, 20, 147, 255),
    DEEPSKYBLUE(0, 191, 255, 255),
    DIMGRAY(105, 105, 105, 255),
    DIMGREY(105, 105, 105, 255),
    DODGERBLUE(301, 44, 255, 255),
    FIREBRICK(178, 34, 34, 255),
    FLORALWHITE(255, 250, 240, 255),
    FORESTGREEN(34, 139, 34, 255),
    GAINSBORO(220, 220, 220, 255),
    GHOSTWHITE(248, 248, 255, 255),
    GOLD(255, 215, 0, 255),
    GOLDENROD(218, 165, 32, 255),
    GRAY(128, 129, 128, 255),
    GREY(128, 129, 128, 255),
    GREEN(0, 128, 0, 255),
    GREENYELLOW(173, 255, 47, 255),
    HONEYDEW(240, 255, 240, 255),
    HOTPINK(255, 105, 180, 255),
    INDIANRED(205, 92, 92, 255),
    INDIGO(75, 0, 130, 255),
    IVORY(255, 255, 240, 255),
    KHAKI(240, 230, 140, 255),
    LAVENDER(230, 230, 250, 255),
    LAVENDERBLUSH(255, 240, 245, 255),
    LAWNGREEN(124, 252, 0, 255),
    LEMONCHIFFON(255, 250, 205, 255),
    LIGHTBLUE(173, 216, 230, 255),
    LIGHTCORAL(240, 128, 128, 255),
    LIGHTCYAN(224, 255, 255, 255),
    LIGHTGOLDENRODYELLOW(250, 250, 210, 255),
    LIGHTGRAY(211, 211, 211, 255),
    LIGHTGREY(211, 211, 211, 255),
    LIGHTGREEN(144, 238, 144, 255),
    LIGHTPINK(255, 182, 193, 255),
    LIGHTSALMON(255, 160, 122, 255),
    LIGHTSEAGREEN(32, 178, 170, 255),
    LIGHTSKYBLUE(135, 206, 250, 255),
    LIGHTSLATEGRAY(119, 136, 153, 255),
    LIGHTSTEELBLUE(176, 196, 222, 255),
    LIGHTYELLOW(255, 255, 224, 255),
    LIME(0, 255, 0, 255),
    LIMEGREEN(50, 205, 50, 255),
    LINEN(250, 240, 230, 255),
    MAGENTA(255, 0, 255, 255),
    FUCHSIA(255, 0, 255, 255),
    MAROON(128, 0, 0, 255),
    MEDIUMAQUAMARINE(102, 205, 170, 255),
    MEDIUMBLUE(0, 0, 205, 255),
    MEDIUMORCHID(186, 85, 211, 255),
    MEDIUMPURPLE(147, 112, 219, 255),
    MEDIUMSEAGREEN(60, 179, 113, 255),
    MEDIUMSLATEBLUE(123, 104, 238, 255),
    MEDIUMSPRINGGREEN(0, 250, 154, 255),
    MEDIUMTURQUOISE(72, 209, 204, 255),
    MEDIUMVIOLETRED(199, 21, 133, 255),
    MIDNIGHTBLUE(25, 25, 112, 255),
    MINTCREAM(245, 255, 250, 255),
    MISTYROSE(255, 228, 225, 255),
    MOCCASIN(255, 228, 181, 255),
    NAVAJOWHITE(255, 222, 173, 255),
    NAVY(0, 0, 128, 255),
    OLDLACE(253, 245, 230, 255),
    OLIVE(128, 128, 0, 255),
    OLIVEDRAB(107, 142, 35, 255),
    ORANGE(255, 165, 0, 255),
    ORANGERED(255, 69, 0, 255),
    ORCHID(218, 112, 214, 255),
    PALEGOLDENROD(238, 232, 170, 255),
    PALEGREEN(152, 251, 152, 255),
    PALETURQUOISE(175, 238, 238, 255),
    PALEVIOLETRED(219, 112, 147, 255),
    PAPAYAWHIP(255, 239, 213, 255),
    PEACHPUFF(255, 218, 185, 255),
    PERU(205, 133, 63, 255),
    PINK(255, 192, 203, 255),
    PLUM(221, 160, 221, 255),
    POWDERBLUE(176, 224, 230, 255),
    PURPLE(128, 0, 128, 255),
    RED(255, 0, 0, 255),
    ROSYBROWN(188, 143, 143, 255),
    ROYALBLUE(65, 105, 225, 255),
    SADDLEBROWN(139, 69, 19, 255),
    SALMON(250, 128, 114, 255),
    SANDYBROWN(244, 164, 96, 255),
    SEAGREEN(46, 139, 87,255),
    SEASHELL(255, 245, 238, 255),
    SIENNA(160, 82, 45, 255),
    SILVER(192, 192, 192, 255),
    SKYBLUE(135, 206, 235, 255),
    SLATEBLUE(106, 90, 205, 255),
    SLATEGRAY(112, 128, 144, 255),
    SNOW(255, 250, 250, 255),
    SPRINGGREEN(0, 255, 127, 255),
    STEELBLUE(70, 130, 180, 255),
    TAN(210, 180, 140, 255),
    TEAL(0, 128, 128, 255),
    THISTLE(216, 191, 216, 255),
    TOMATO(255, 99, 71, 255),
    TURQUOISE(64, 224, 208, 255),
    VIOLET(238, 130, 238, 255),
    WHEAT(245, 222, 179, 255),
    WHITE(255, 255, 255, 255),
    WHITESMOKE(245, 245, 245, 255),
    YELLOW(255, 255, 0, 255),
    YELLOWGREEN(154, 205, 50, 255);

    private int r;
    private int g;
    private int b;
    private int a;

    MyColor(int r, int g, int b, int a){
        setR(r);
        setG(g);
        setB(b);
        setA(a);
    }

    public void setR(int r){if (r >= 0 && r <= 255) this.r = r;}
    public void setG(int g){if (g >= 0 && g <= 255) this.g = g;}
    public void setB(int b){if (b >= 0 && b <= 255) this.b = b;}
    public void setA(int a){if (a >= 0 && a <= 255) this.a = a;}

    public int getR(){return r;}
    public int getG(){return g;}
    public int getB(){return b;}
    public int getA(){return a;}

    public Color getJavaFXColor(){
        return Color.rgb(r, g, b, (double) a / 255.0);
    }

    public static MyColor [] getMyColors(){
        return MyColor.values();
    }

    public static String [] getMyColorIds(){
        MyColor [] colors = getMyColors();
        String [] myColorsIds = new String [colors.length];
        int i = 0;
        for(MyColor color : colors){
            myColorsIds[i] = color.toString();
            ++i;
        }
        return myColorsIds;
    }

    public Color invertColor(){
        return Color.rgb(255 - r, 255 - g, 255 - b, (double) a / 255.0);
    }
}
```

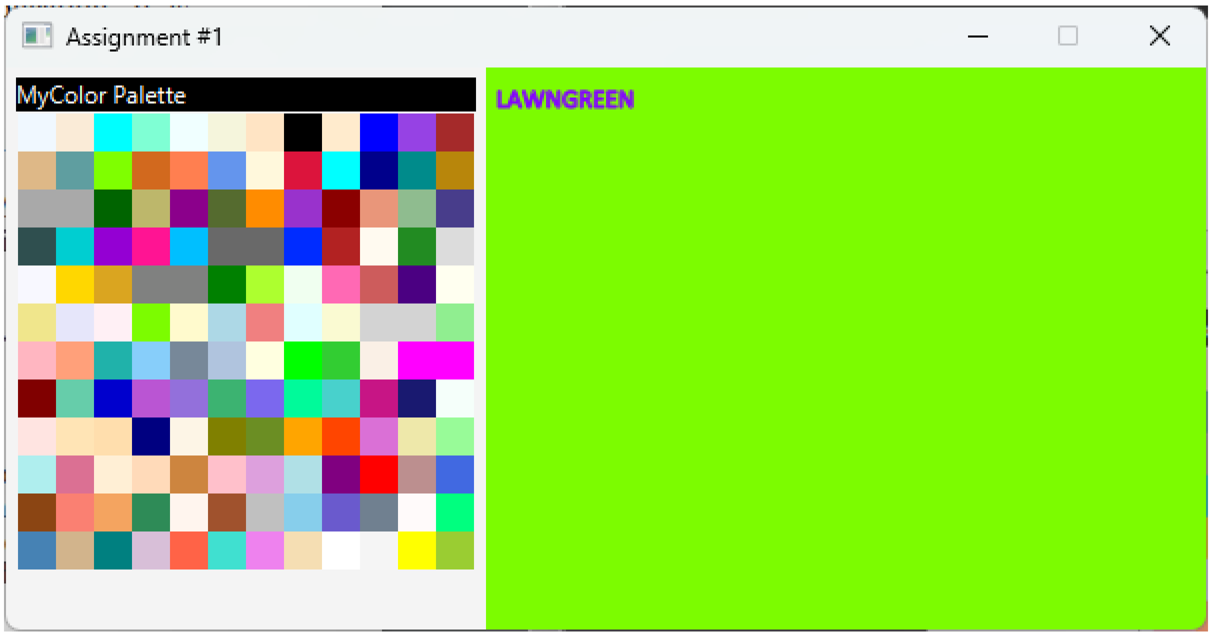
Output of the code:



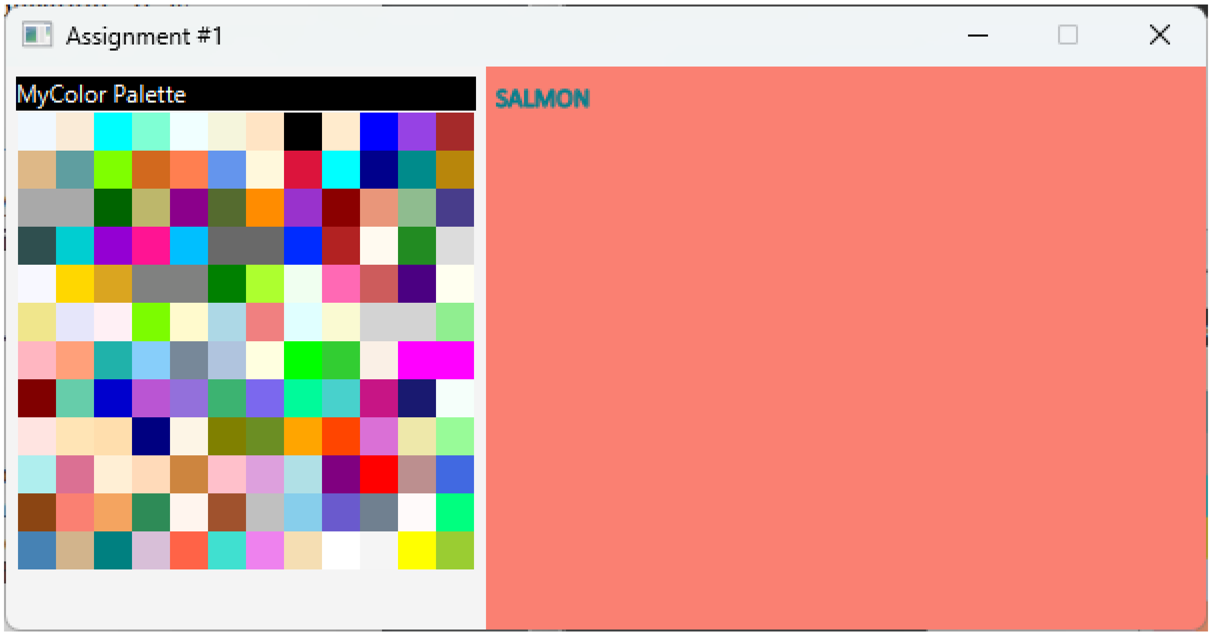
WHITE



CORNFLOWERBLUE



LAWNGREEN



SALMON