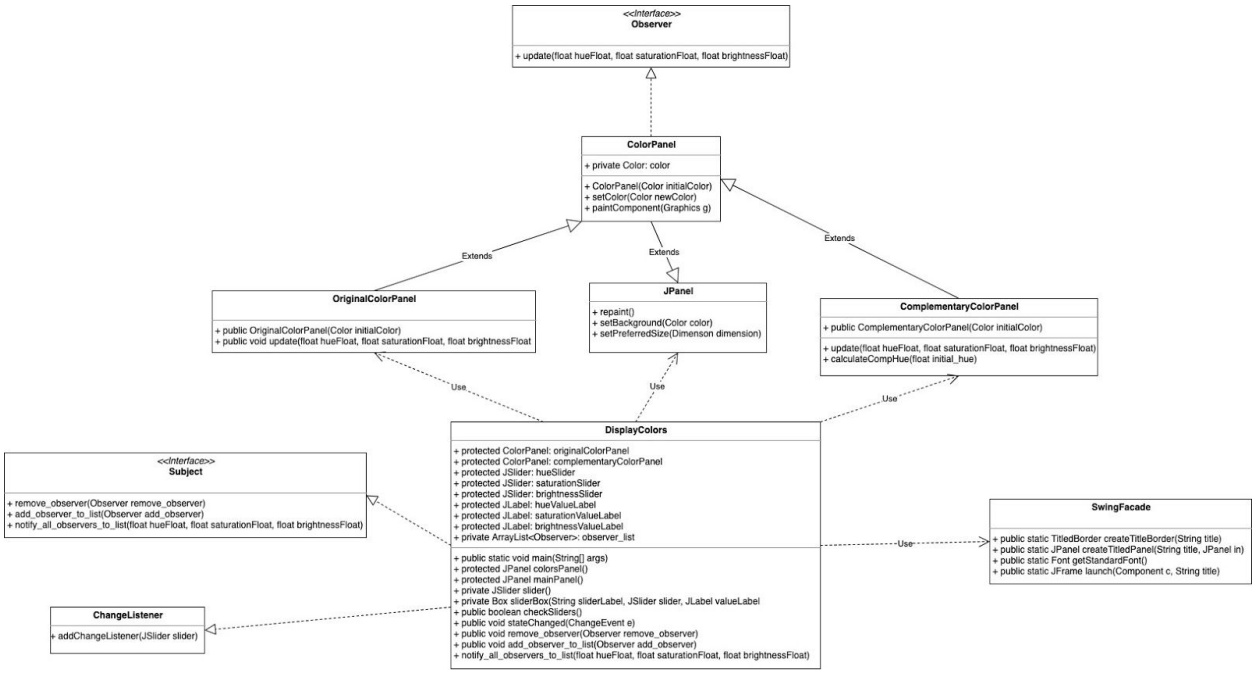
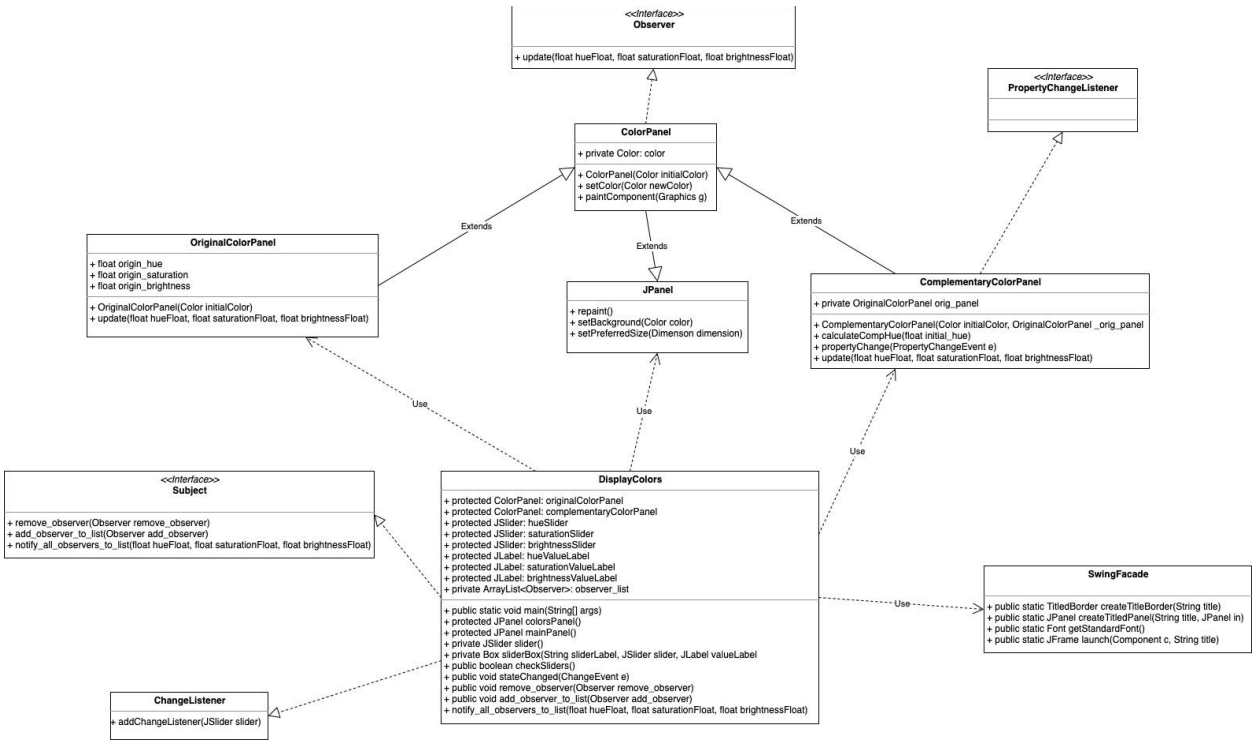
Andrew Brown
EE461L
11/24/20

Tutorial #10 Report

Task 2 Class Diagram:

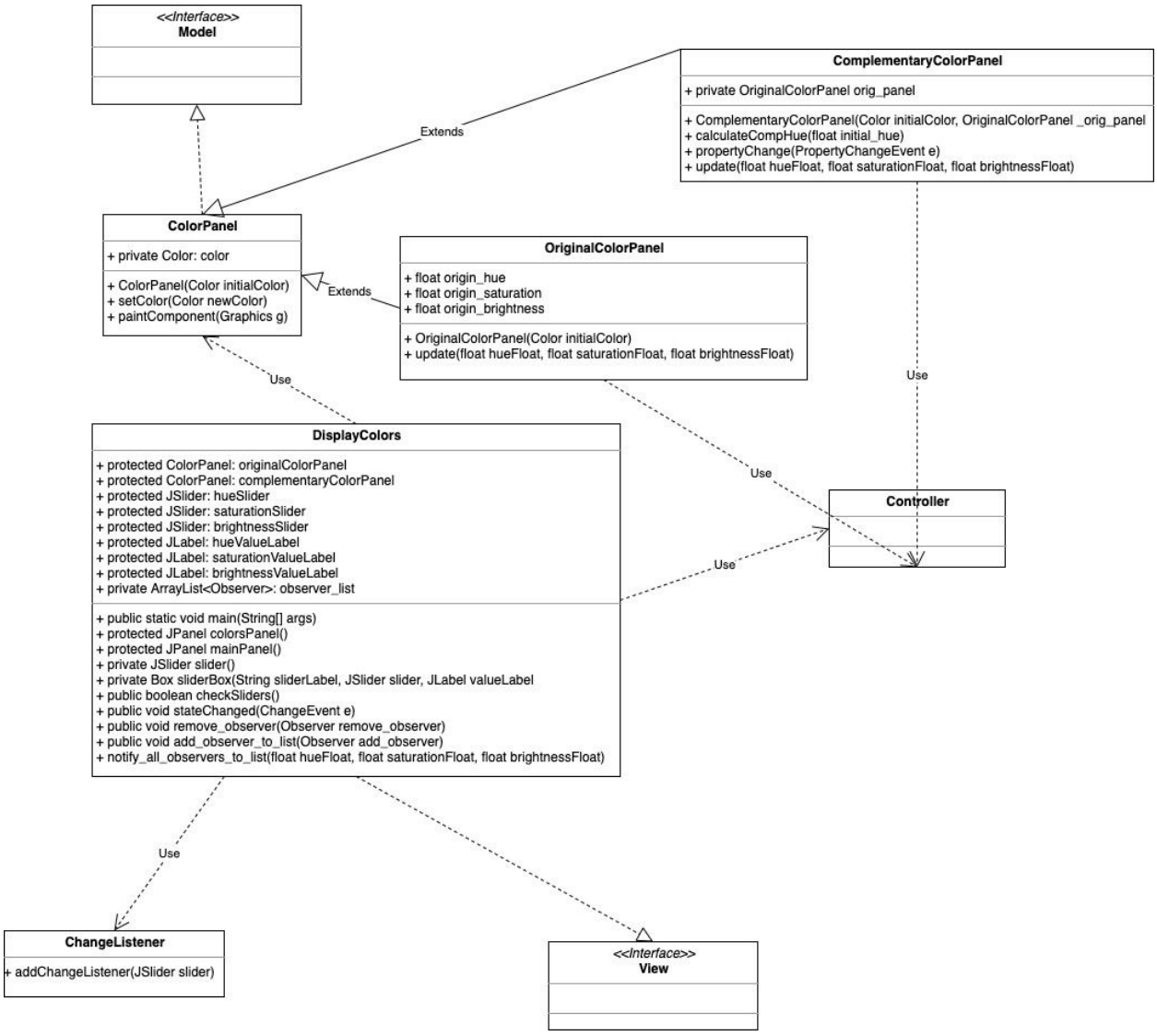


Task 3 Class Diagram:



Task 4:

The Model View Controller (MVC) design pattern specifies that an application consist of a data model, presentation information, and control information. The pattern requires that each of these be separated into different objects. The data model solely contains the pure application data. For Tutorial #10, the model would correspond to the colors. The view presents the model's data to the user. For Tutorial #10, the view would correspond to the DisplayColors module. DisplayColors allows the users to interact with the sliders and the color panels. The controller listens to events triggered by the view and executes the appropriate reaction to these events. For Tutorial #10, we do not have a defined controller already implemented. Therefore, we will have to create a specific controller class that implements the ChangeListener interface and thus communicate and update the view component whenever the user inputs new data. The link to MVC source is listed here: https://www.geeksforgeeks.org/mvc-design-pattern/


<<Interface>>
Model
ComplementaryColorPanel
+ private OriginalColorPanel orig_panel
+ ComplementaryColorPanel(Color initialColor, OriginalColorPanel _orig_panel
+ calculateCompHue(float initial_hue)
+ propertyChange(PropertyChangeEvent e)
+ update(float hueFloat, float saturationFloat, float brightnessFloat)
Extends
ColorPanel
+ private Color: color
+ ColorPanel(Color initialColor)
+ setColor(Color newColor)
+ paintComponent(Graphics g)
Extends
OriginalColorPanel
+ float origin_hue
+ float origin_saturation
+ float origin_brightness
+ OriginalColorPanel(Color initialColor)
+ update(float hueFloat, float saturationFloat, float brightnessFloat)
Use
Use
Use
Use
DisplayColors
+ protected ColorPanel: originalColorPanel
+ protected ColorPanel: complementaryColorPanel
+ protected JSlider: hueSlider
+ protected JSlider: saturationSlider
+ protected JSlider: brightnessSlider
+ protected JLabel: hueValueLabel
+ protected JLabel: saturationValueLabel
+ protected JLabel: brightnessValueLabel
+ private ArrayList<Observer>: observer_list
+ public static void main(String[] args)
+ protected JPanel colorsPanel()
+ protected JPanel mainPanel()
+ private JSlider slider()
+ private Box sliderBox(String sliderLabel, JSlider slider, JLabel valueLabel
+ public boolean checkSliders()
+ public void stateChanged(ChangeEvent e)
+ public void remove_observer(Observer remove_observer)
+ public void add_observer_to_list(Observer add_observer)
+ notify_all_observers_to_list(float hueFloat, float saturationFloat, float brightnessFloat)
Controller
Use
Use
ChangeListener
+ addChangeListener(JSlider slider)
<<Interface>>
View