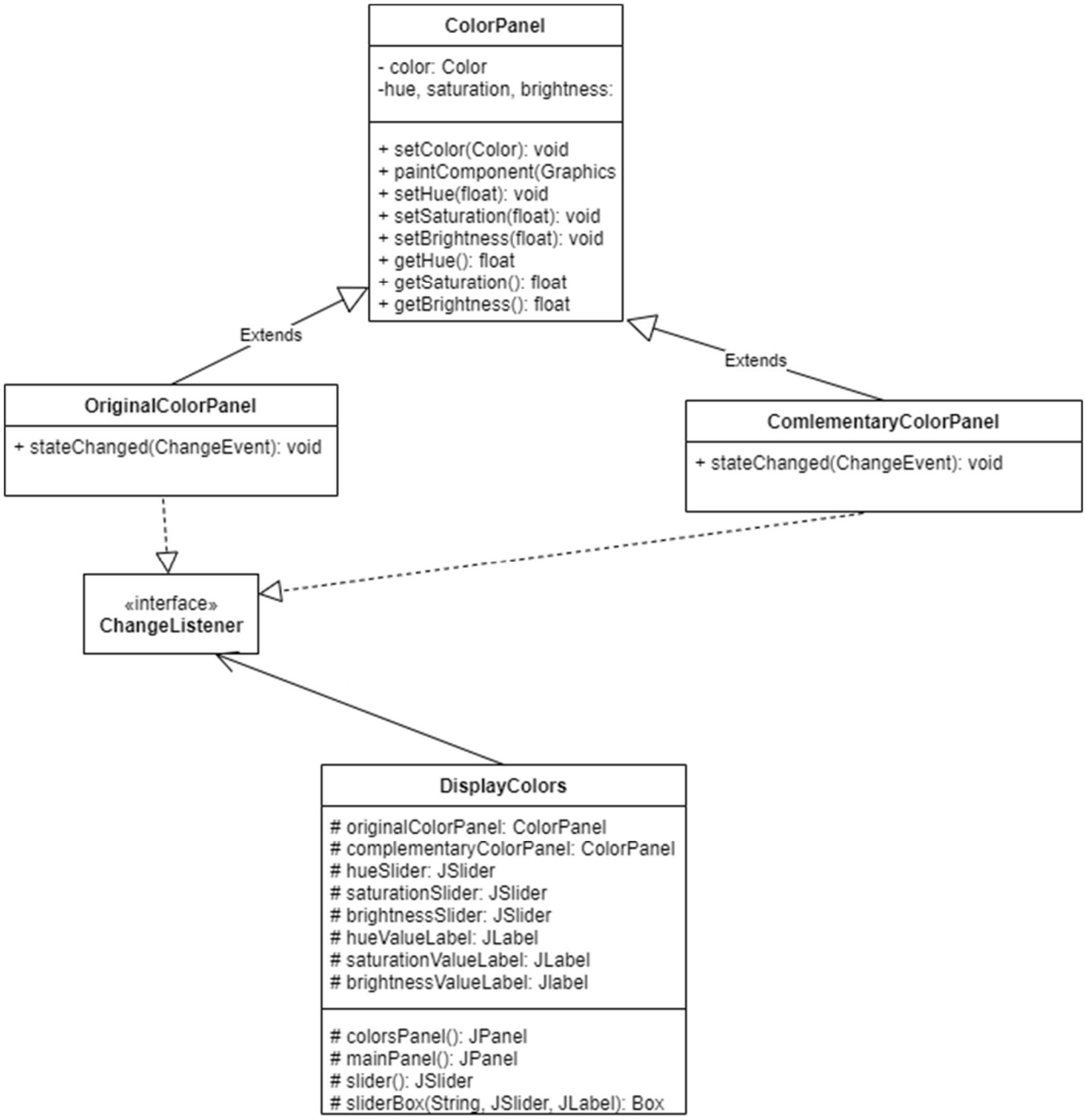
EE461L Tutorial 10

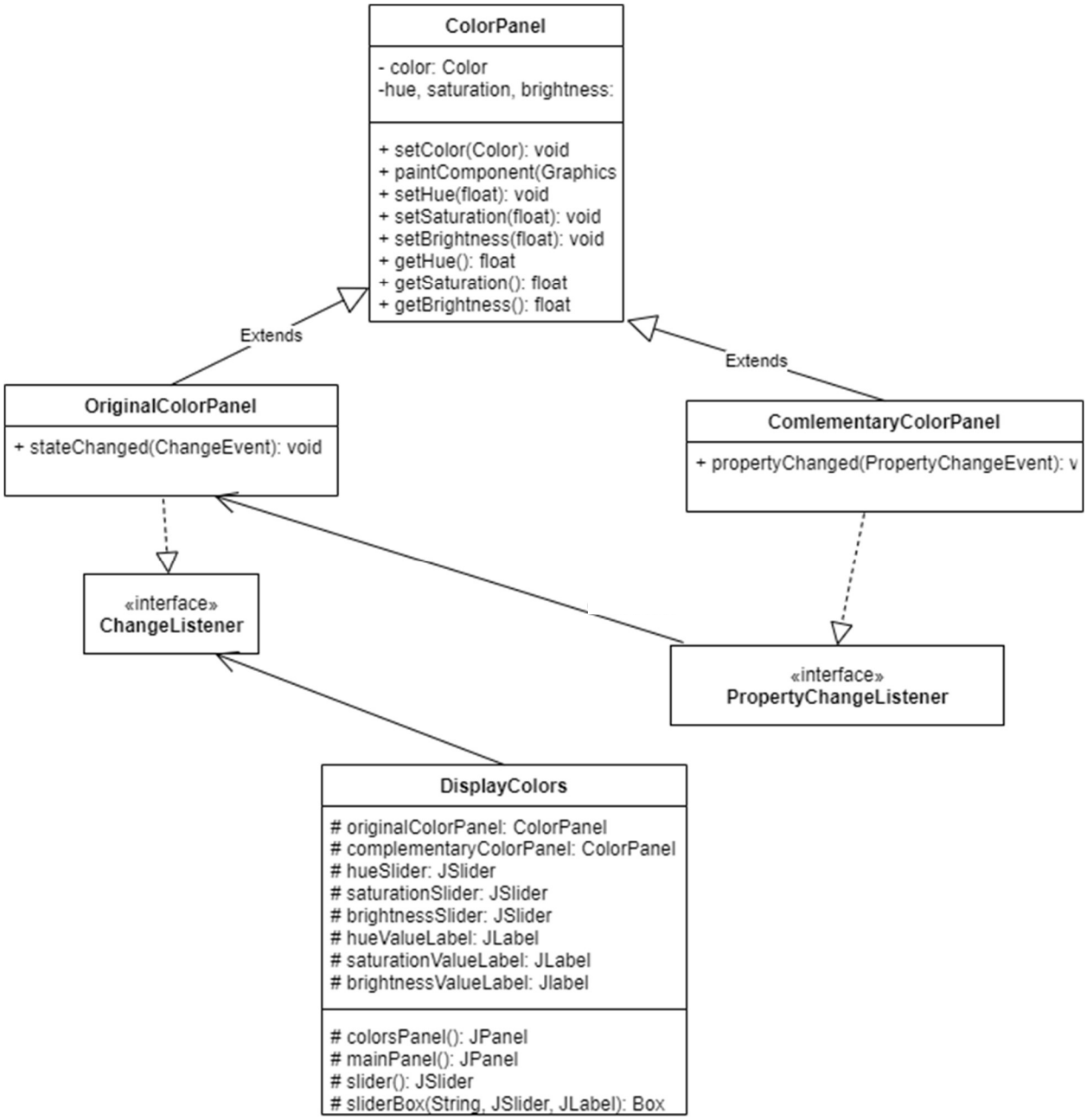
Tyler Pak

tjp2365

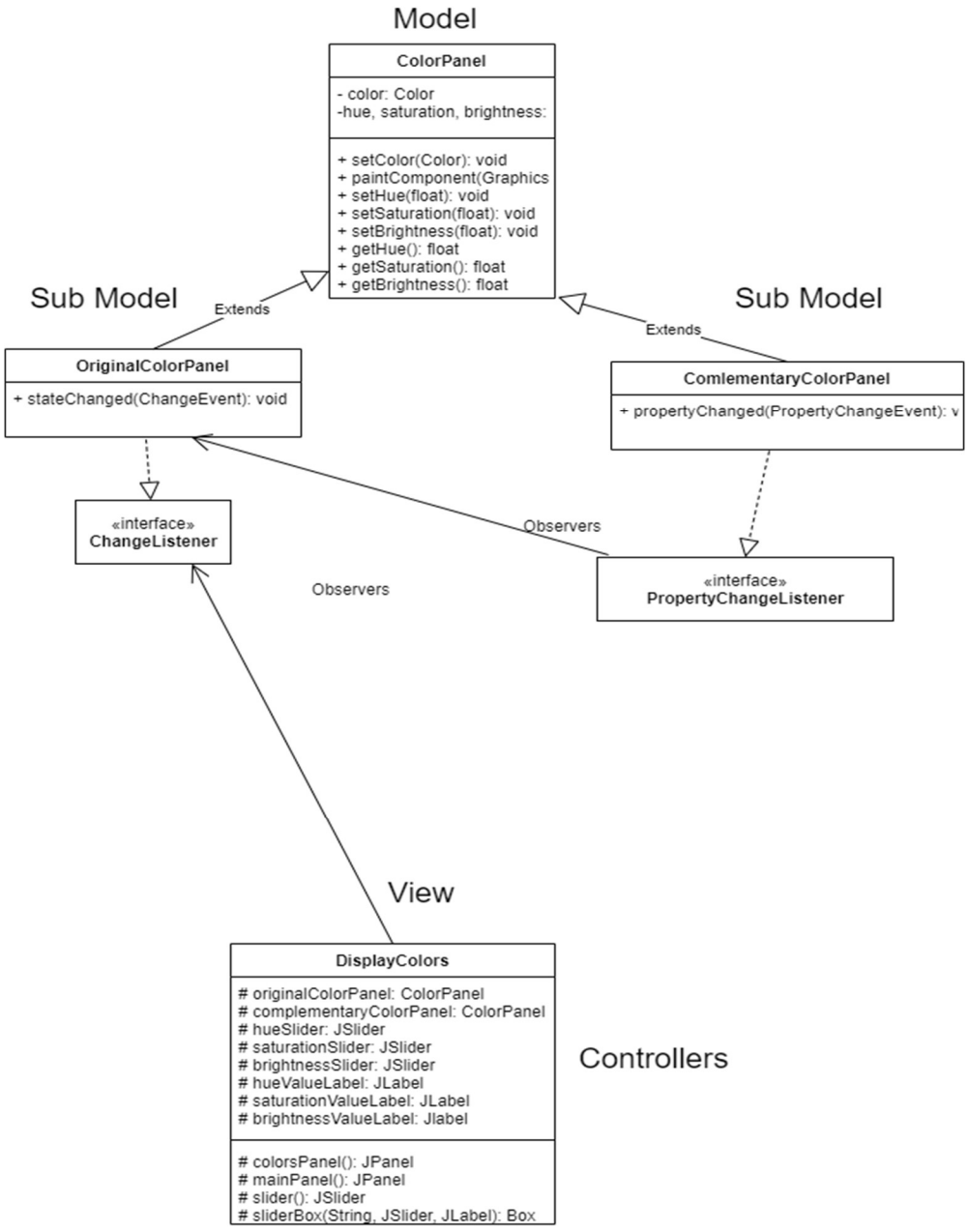
Task 2:



Task 3:


ColorPanel
- color: Color
-hue, saturation, brightness:
+ setColor(Color): void
+ paintComponent(Graphics
+ setHue(float): void
+ setSaturation(float): void
+ setBrightness(float): void
+ getHue(): float
+ getSaturation(): float
+ getBrightness(): float
Extends
Extends
OriginalColorPanel
+ stateChanged(ChangeEvent): void
ComlementaryColorPanel
+ propertyChanged(PropertyChangeEvent): v
«interface»
ChangeListener
«interface»
PropertyChangeListener
DisplayColors
originalColorPanel: ColorPanel
complementaryColorPanel: ColorPanel
hueSlider: JSlider
saturationSlider: JSlider
brightnessSlider: JSlider
hueValueLabel: JLabel
saturationValueLabel: JLabel
brightnessValueLabel: Jlabel
colorsPanel(): JPanel
mainPanel(): JPanel
slider(): JSlider
sliderBox(String, JSlider, JLabel): Box

Task 4:

I could refactor this program into an MVC design pattern by setting the model as color panel, and original and complementary color panels as sub models. I would set the DisplayColors class to view and the controller. The observers in this pattern would be the complementary color panel and the displaycolors classes. The complementary color panel would observe the original color panel for change. Display Colors observes the original color panel.

Anytime there is a change in the sliders, DisplayColors would observe this and change the submodel OriginalColorPanel, which the ComplementaryColorPanel would observe.