Matthew Zhan, myz84

3/25/16

EE 461L Spring 2016

Tutorial 7

A. Task 1

The diagram below shows the relationship between ColorPanel and DisplayColors and the interface ChangeListener implemented by DisplayColors.

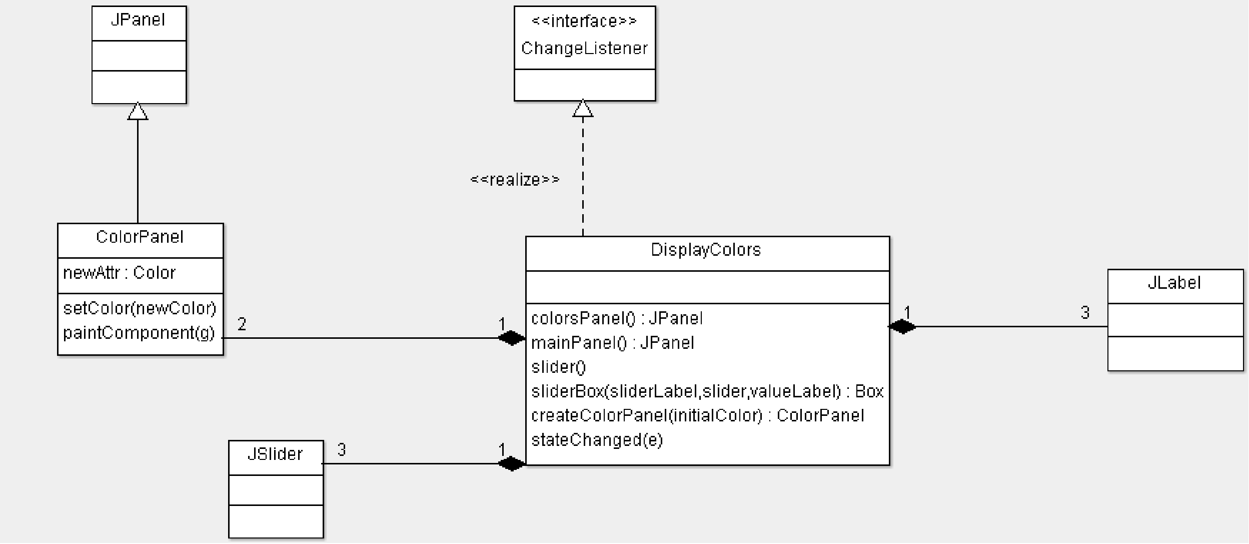

Figure 1. *Class Diagram of ColorPanel and DisplayColors*

B. Task 2

The diagram below shows the change of moving the ChangeListener to the ColorPanel Object and two subclasses for ColorPanel to implement their own stateChanged(e) function.

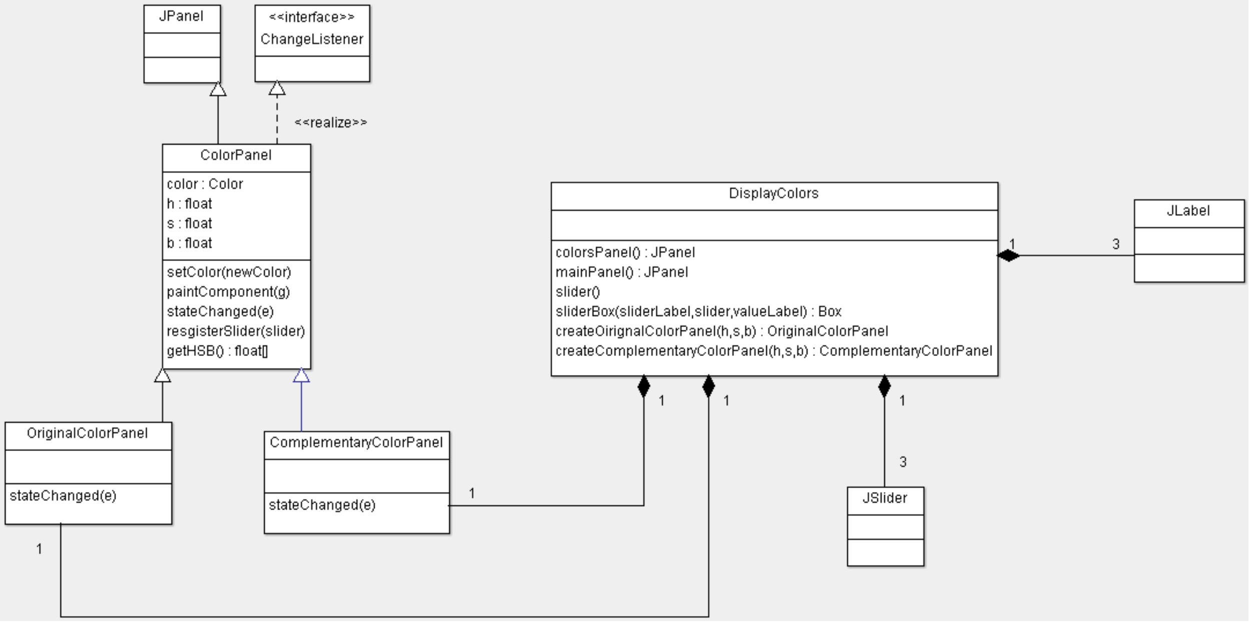

Figure 2. Class Diagram of OriginalColorPanel and ComplementaryColorPanel

C. Task 3

The diagram below shows the change of ComplementaryColorPanel implementing a PropertyChangeListener to update its color when the OriginalColorPanel is changed, instead of using the sliders to update its color.

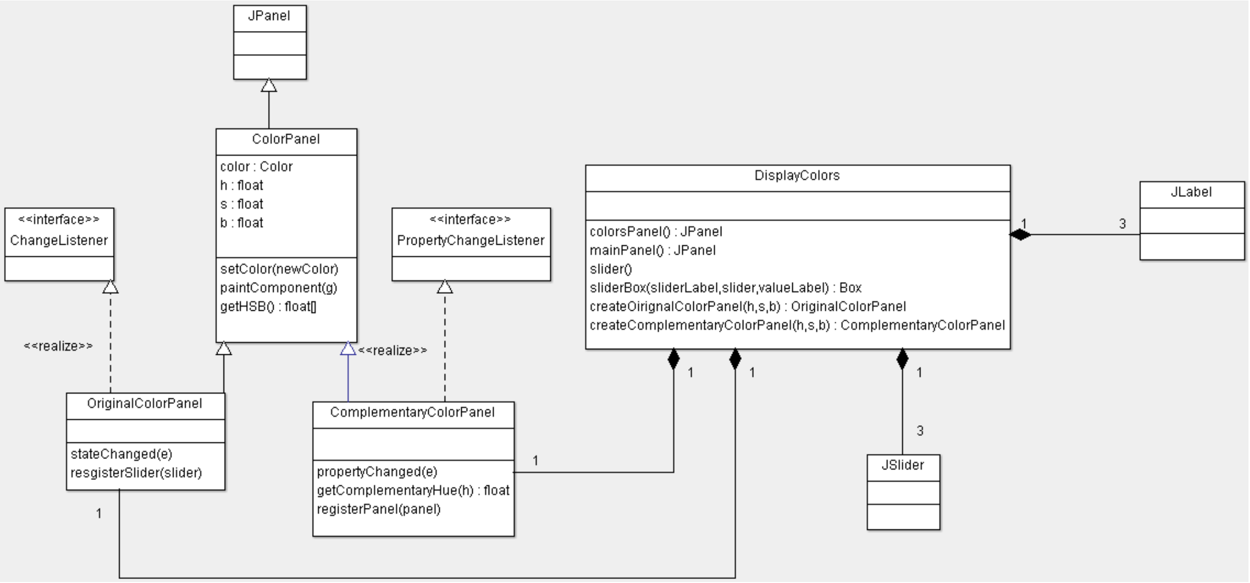

Figure 3. Class Diagram of ComplementaryColorPanel implementing PropertyChangeListener

D. Task 4

The Model View Controller design pattern is relevant to our problem because each component in the design can be filled by a component in the problem. In particular, the color is our observable object, which is observed by the color panels and manipulated by the sliders. The model directly manages the data of application and we are concerned with updating and reporting one Color. A view is any representation of information, in our case the current color. Lastly, the controller accepts input from the user and manipulates the model or view, in our case the sliders manipulate the color data (https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller).

In this way, the sliders or controllers directly manipulate the Color in the model. The Color is the observable object and the ColorPanel is the observer. So whenever the Color is changed, Panel objects will be updated and the view will change based on the Model. The controller and view would both be displayed on the same application window.

The design below shows an Application consisting of one ColorModel, three sliders, and two ColorPanels. The three slider controllers will trigger stateChanged in the Application, which then updates MyColor, the observable Color object in the ColorModel. The ColorView and Complementary ColorView observe MyColor and are updated when MyColor changes.

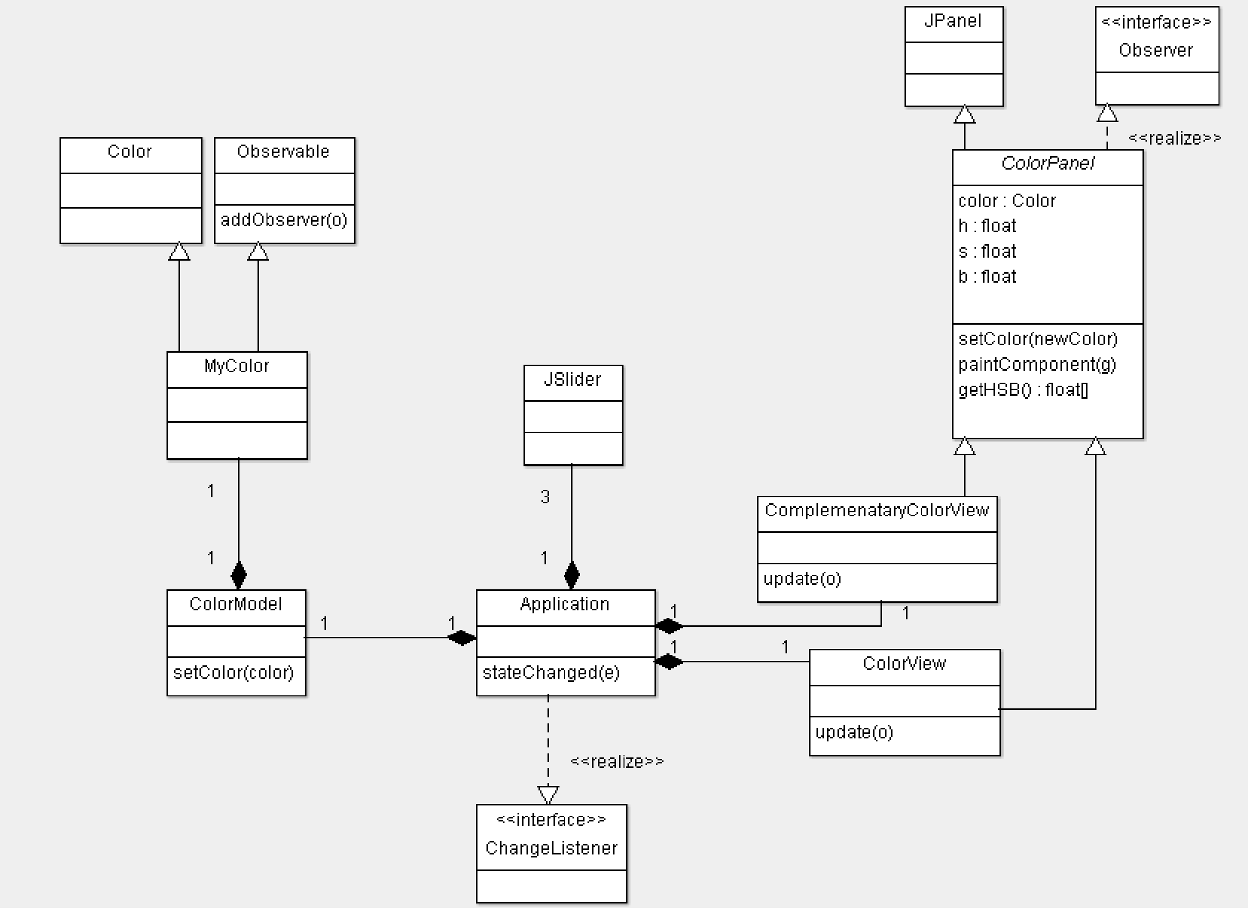

Figure 4. *Class Diagram using the MVC Design Pattern*