Rohit Krishnan
Gordon Liang
Kareem Khattab
Brian Lee

Assignment 4: Write Up

For this assignment we utilized the Observer & Command patterns. We utilized their strengths while discarding their weaknesses. We implemented the Observer pattern by assigning each use case or "event" with a corresponding action. We carefully maintained this structure, so it is easier for us to go in and make changes, such as adding an event & action. For example, we treat each user input as an event, and each event has a corresponding action. We maintained a list of these dependencies, which are essentially observers, and code that notifies and updates the event. Because the Observer patter is essentially a deviation of the MVC(Model, View, Controller), it is pretty much implied that we will implementing the pattern in some form when dealing with a User Interface/GUI where the user can perform certain actions on the GUI, and based on these actions, the backend is updated. We have a simple view called showGUI.java that contains the main method and where the all the logic & GUI formatting is visualized. Menu.java is essentially the controller, as it contains the actionPerformed code that dictates what happens when the user performs a certain action.

We implemented the Command pattern in numerous occasions. Most notably is our Draggable.java class. This class is separate & can be applied to any JComponent and turn that component into a component that the user is able to drag around and drop, using his/her mouse. We utilized a client class that contains the decision making, on the various use cases that we were supposed to implement into the overall program. Our Image.java class is another good example. It code that displays an image, and is then called by the "invoker". Again, I feel it is implied that when one is creating a program of this nature, these two patterns (Observer & Command) will be utilized. Often I've caught myself writing a program, especially GUI programs, with no specific intention of implementing those patterns, but once it's all said and done, and I take a look at the codebase, I notice that I have utilized both patterns. However, this time around, for the sake of academia, we made a conscious effort to make sure these patterns were implemented "correctly" without messing up our own personal coding personality & style.