David Watanabe

Java Project Narrative: Homework Planner

Before I started to design or code anything, I knew that a major part of my project would include at least two packages that we have not yet learned as part of AP Computer Science. The first one was Swing, which would be used to handle the GUI. The other one was I/O, which would be used to save and read information. Although I have worked with both of these before in past projects outside of class, it would still be a sufficient challenge for me as a coder. This time I had to make the program write information to a text file while previously I only had it read information. Furthermore, it was over half a year ago since I last used Swing and I/O, so I was a bit out of practice. This is why, during the first week, I spent nearly all of my reading instead of coding. (Record of Thinking #1)

I focused on reading the documentation for Swing, so I would have a rough idea of the capabilities of the JComponents. From there, I decided how the user would interact with the program. (JRadioButtons, JButtons, JComboBox, etc.) I also spent time looking at my past projects as a refresher. Looking at those helped to a certain extent, but not as much as I would've liked because some dimwit didn't comment his code.

The first class that I wrote was the driver class (Launcher). I got it done fairly quickly since all what it does is call upon the Interface class. My task for the second week was to create the interface. The biggest challenge with coding the GUI from the ground up is the mental translation from a visual image into code. Just like how a picture is worth a thousand words, I had to describe the appearance of a simple interface with many lines of code. Even with the assistance of a diagram, it still took a lot of effort to understand the functionality of different layouts (i.e. GridBagLayout, FlowLayout, and GridLayout) and determine which one is appropriate for the design. One problem that I faced was displaying the assignments. I decided to use radio buttons because they would allow the user to select and interact with specific assignments when used in conjunction with buttons. However, I couldn't manually create each of these radio buttons because the number of assignments is unspecified. Instead I had to create a for loop that would add radio buttons to an ArrayList whenever the user creates a new assignment.

The last week of the project was quite scary for me. By that point I had the interface finished, but nothing else. Basically all I had was buttons that didn't do anything. I needed to make some pretty quick decisions on how to go about doing the TextScanner. Thankfully, I was able use a portion of code from a previous project to get it to read from a text file. In terms of writing to a text file, I had read of solutions on how to append text to the end of the text file, but I couldn't find much on how to insert text into the middle of a text file (Apparently it is not advisable to do that because it can cause some complications if the text file gets accidentally changed by the user). I decided to take the approach where I re-wrote the entire text file each time there was a modification to the data. (March 7 Commit) This way was the simplest to deal with since the user would be able to delete an assignment from the middle of the list.

After the program reads the data from text, I had it save to an array list that would be used by the Format class. Format was probably one of the easier classes to write because it primarily handles string manipulation – something that we've covered in AP comp sci.

I had a small amount of collaboration with other people. I made a few suggestions to Aliya to help her in regards to Swing. I also attempted help Brianna and Troy with sorting out some bugs (though I wasn't successful with either of them). The beta test was very helpful. By watching Troy I was able to find out that marking assignments complete/incomplete wasn't immediately obvious in the UI, so I opted to make it so that the font changed to have a strikethrough. (March 8 Commit) I also followed Troy's suggestion to add a warning to the delete button. If I had more time to work on the project, I would've implemented a method that could sort assignments by date.

If I could do something differently, I would've been more liberal about committing my code. I often hesitated about committing because I almost never had entirely working code at any point. Looking back, I was putting myself at a huge risk because if something happened to my computer, I wouldn't have been able to recover my work. Not having commit messages also made writing this reflection a bit difficult.