Code Review Description

Overall, the codebase is less than ideal. It took the three of us roughly three hours to go through it in its entirety. During this processed we mapped out which classes use other classes, and the final mapping was spaghetti. Before we dive deeper into the negatives let’s highlight the positives.

The 88 files were organized into 16 different packages. This was great for knowing where files were located and what behaviors they were a part of. The UI for the most part was functional albeit it is tedious to use. Additionally, there are a few exceptions that are unhandled. The code is extremely readable. The code is also well commented. These two aspects make the code much easier to understand. They did follow the style guide. There was only one large class: the sprite class. Lastly, we think there is good O.O.P. design. And the classes are well encapsulated and separated.

After we mapped out the classes, we concluded there was no clear understanding of MVC. There was no clear View, Controller, and Model classes. Each responsibility was spread out among multiple classes, which is reasonable. What is not reasonable is that some classes did a combination of view, application, and business logic thus mixing the responsibilities of the View, Controller, and Model.

Overall, a majority of the patterns were used correctly. The design patterns that were used correctly were Composite, Strategy, and Observer. Although, it’s possible there is an over usage of the Strategy Pattern. And Observable is not an interface which is a red flag. MVC was not used correctly as described above. Command Pattern was not used at all but there were Command files in the codebase that were not used.

There were some files that were not used at all. As well as some files that were completely missed name. Both of these aspects gave us a difficult time understanding the codebase. We inherited the codebase with 179 warnings that we reduced to 9. There were comments that revealed shortcuts were taken. We believe there was a big picture in mind, but it is clear it is only one person’s big picture. If all the teammates worked together it is possible some of the flaws of this codebase could have been avoided. No magic numbers were present in the code. But they have a constants file presumably to hold constant variables. Unfortunately, it was rarely used, and many constant variables were housed in the classes that used them. We cannot say for sure if it is good Java because we are not particularly skilled with using Java.

In terms of documentation and guides there were none. The repository had a ReadMe file with what looked like instructions to run the program that was recycled since Assignment 1. In terms of tools, not many were used. We assume Eclipse was used as the IDE. We know for sure Maven, Git, and Jira were used. Confluence was used but all that was added were the deliverable files. We know for sure Log4J and JUnit were not used. Although JUnit was imported into the project. We do not know if EclEmma, Bamboo, or SonarQube were used.