WordFisher sprint report

2018/02/26 Sprint 1

The plan for the sprint 1 was to first set up a basic structure with minimum code content inside to use as a guide point to how to advance the plug-in towards the next step. By focusing work on the framework and having less content, we'll prevent a struggle happening later on, with being unfamiliar what to code in the content.

We managed to accomplish the first half of the goal. Our main class that will be needed for the plug -in, which was the “searchFile” class and “Target” class, were both finished. “searchFile” will find the selected string in the chosen file, and the “Target” will contain the string which the user what to replace with and the string that will become replaced in the selected file. These two class will become the main core to start writing the test code, and to work on the main document. However, we were not able to progress any farther, and the writing for the “control” “view” “WordFisher” will be have to be worked on the next sprint

The problem we have encountered was a conflict of splitting workload, which lead the start of the work on the sprint to delay. We have tried to overcome by contacting through mail and discussing what would need to be worked first. We managed to start and able to do some work on the sprint, yet the delay of the start has caused to many parts to be left uncompleted

For the next sprint, we will first work on creating a test, running to see the current search file works, and using the result to plan and write in the main code. With the outcome of the test, we will be able to list the section of the code which may cause an error and to fix. Once by testing and figuring out the quality of what we have right now, we will proceed to the actual plug-in parts, which will be working on the view and the controller first. Once the connection of the view, controller and the working class is made we will conclude to work on the main file to have a runnable plug-in

With the next sprint, we will plan forwardly and split work in way where it’s easy to arrange in case of any conflicts occurred during the sprint, so in case where one of us encountered a problem and figure there work speed will decrease, the other will be able to support. We will also try to start more faster then what we did in this sprint so to have a continuous cycle of working and checking to progress, not leaving everything to the last minute.

2018/03/18 Sprint 2

The plan for sprint 2 was to add in the features that we were not able to implement in our first sprint. Since we were able to create a simple working plugin in the first step, the task for this sprint to increase the capability of this plugin, so it would become more user friendly. The features were “tracking change history”, “window class to create input boxes”, “checking multiple files” and having a test class to prefer the plugin works.

Compared to the first sprint, we were able to complete many of the tasks we set out to do. We were able to create a **History class** to contain all the changes that the user will do and which allows us to print the past changes that the user has made. We were also able to create a textbox where the user enters the text to allow the user to choose which text should be replaced and what text will replace in that spot. This feature will allow the user to decide the text they want to replace and increase the usability of the program with a major step, instead of just replacing “test” to “replaced” as in our sprint 1 example. There is also a test class as well, which is currently able to look at some classes to run a unit test for those classes. However, our test function only has the ability for a unit test right now, and not able to run a mock test to check the whole program to run. We also were not able to implement the ability to choose multiple files to be inputted at once, however, due to the loss of a team member, we may opt to cut this feature out of the program entirely depending on how the third sprint goes.

The main conflict that we had in this sprint was the lack in communication and cooperation among teammates. Along the sprint we were able to work on our individual tasks, but we had a hard time on supporting the other teammates work due to poor documentation, which made it difficult for other team members to understand our code. This caused our work process to be done as individuals instead of working on one task as a team at the same time. Being unfamiliar with the other teammates work caused delays in some of our work.

For the next sprint, we will first finish off the tasks that we have not able to complete in this sprint. This will be finishing off the test and having the plugin to be able to implement multiple files at oncee. We also plan to reorganize the GUI, making it look much more professional and well done. Once that is finished, we plan to increase the usability will include to allow a user to undo recent changes made, and possible being able to select from having the change done on the whole file or some parts of the file. Also, a help command to display how to use the plugin, and other user-friendly features that might be necessary.

For our next sprint, we will try to increase interest towards our teammates work, to understand what we can do to help in case we encounter a problem. We will also discuss in what shape the added feature will be implemented as. Have a common understanding for each of our work to increase the quality of our work, and will allow helping to each other to be more easy and not require the process of understanding what function is doing what before helping. This can easily be done by including more documentation.