CS 361 Project 4 Report

Yi Feng, Iris Lian, Chris Marcello, and Evan Savillo

October 3, 2018

Overview

In this week's project, we improved our IDE application from last week by replacing the *TextAreas* with *CodeAreas* (placed inside of *VirtualizedScrollPanes)*, which enabled us to add useful features such as line numbers colored keywords.

Breakdown

In this project, we have these files: Main.java, Main.fxml, ColoredCodeArea.java, UserErrorDialog.java, java-keywords.css, Controller.java, FileMenuController.java, and EditMenuController.java. Main.java starts the program by loading Main.fxml and creating the main Controller. The main controller creates FileMenuController and EditMenuController and delegates tasks to them, which in turn use ColoredCodeArea and UserErrorDialog as helper classes.

Elegance

We created the ColoredCodeArea class to contain the code for implementations of the CodeArea. This class handles all aspects of the CodeArea, including keyword matching, color-coding, and creation of the areas themselves. This allows us to simplify the code significantly by keeping all the code for the CodeArea self-contained and divided from the rest of the code.

Another elegant addition to our code is the *UserErrorDialog* class. Occasionally, our IDE will have to display error messages to the user, and although each message must be different depending with the type of error provoked, the code to write these are rather boilerplate. Thus, the answer is a custom class, *UserErrorDialog,* which extends *Alert* and carries inside of it an *enum ErrorType* definition which handles the particularities of the messages it needs to display. *ErrorType* also makes it obvious in the code what kind of error is being provoked. This class will prove very useful as the future becomes the present: although now the only error messages that need to be displayed to the user are *read-* and *save-*type errors, there will likely be many more as we add functionality to our IDE, and handling new cases will be as simple as defining another *ErrorType*.

Our main addition was the division of the labor of the main controller, Controller.java, into FileMenuController.java and EditMenuController.java. We left Controller.java as the intermediary because of the limitations of FXML, which doesn't permit loading multiple controllers within a single FXML file. This slight inelegance of having an intermediary is more than made up for by the improved organization of the overall code base. Controller.java was previously nearing seven hundred lines of code, so dividing it into two parts with clear purposes has significantly improved the readability of our code.

Division of Work

- Iris and Yi worked on the CodedArea
- Chris and Evan worked on the menu items
- We worked on dividing the controller together
- We debugged and checked the code for elegancy together