Names: Iris Lian, Zeb Keith-Hardy, Michael Li
Course: CS461
Date: March 21, 2019

Project 15: Connect the IDE with the Mars assembler and interpreter

Overview

In this project, we aim to include Mars assembler and interpreter into our IDE. By including the Mars jar file in our project, we are able to assemble and run MIPS files. To enable the highlighting, we also created a MIPSCodeArea class along with a MIPSStyle inner class to create the style that looks like the one in the Mars IDE. The IDE checks the file type and displays different highlighting correspondingly. Moreover, the buttons are enabled/disabled based on the current file.

Details

Syntax Highlighting

- We created a MIPSCodeArea class with a MIPSStype inner class. It has the same structure as the JavaCodeArea class. We added the operations, directives, and registers to the style and updated the regexs so that we can highlight the syntax correctly

Buttons and Assembler Using Mars

- We were able to use most of our code from Project 5 to implement the assembler for this project. Each time we press the assemble or assemble and run button, we create a new thread where we create a process to run the JAR file in headless mode. Using our old code, we stream our input and output to and from the MARS simulator to the console so that we can interact with the GUI. Pressing the buttons also appropriately enables the Stop button and disables the assemble buttons while the thread is active

Elegance

- We properly disable toolbar buttons when you switch between a bantam file and a mips file, and disable the toolbar buttons when you open a file that cannot be assembled or compiled

- We made it so that the highlighting works correctly not only when the user opens files but also when the user creates new files and saves them. In the new tab, the default highlighting is the Bantam Java style. When the file is saved, the highlighting style changes corresponding to the file extension given by the user.

Inelegance

- enableCorrectButton is called twice in the case of handleOpen. This is a necessary bug because enableCorrectButton is connected to a listener when the tab changes, but the filename has not been declared by the time the handleOpen creates a new tab. So we had to call enableCorrectButton again in the handleOpen function.
- We discovered a bug where if we open a Block comment without closing it in the GUI, we get a stack overflow error having to do with the patterns. We checked with Dale and he told us to submit the code as is because he could not reproduce the error.

Division of Work

Zeb took the code from project 5 and refactored it to work for the assembler, Iris did the syntax highlighting, Michael did the work on getting buttons to enable and disable depending on what is running in the GUI, as well as enabling and disabling buttons depending on the type of file open in the current tab. We were all responsible for going through our code and making sure that it is as bug free as possible and making sure that it is properly commented.