CMPT 440 Milestone

Alexander Goncalves

Prof Rivas
CMPT 440
4/2/2017

Abstract

The purpose of this paper is to explain my work so far. I intend to inform on the progress of my project as well as the occurrences while implementing it. In addition to this, I will include some information on how the application works as well as my graphical depiction of the DFA too.

Introduction

My project is a small application that mimics terminal interfaces similar to Bash and PowerShell. It takes inspiration for command names explicitly from Linux. The interesting element is that this interface uses a deterministic finite automata for the entry and acceptance of it's commands. The DFA implementation will be done via table upon completion. This project is done entirely using Java as well as Spring framework to handle user interface. I thought this was an interesting project idea which would give me an opportunity to try something new. Currently I have begin some work on the system and would gauge myself to be 15% complete with the project. I hope to have the project complete within the upcoming weeks.

Detailed System Description

So far, the system has several different components, that are independent for the time being. I have the UI set up for the most part, which includes an identical look and behavior to many existing shells. Currently it doesn't actually interface with the functionality yet, as that connection will probably be the final step. In addition to this, I have a directory with my existing command implementations: whoami and mkdir. Both of which mimic the classic Linux commands, printing the user and creating a directory respectively. Next, is the DFA which has an instance of both of these commands. In the DFA file I have not yet implemented the table for the sole reason that it would constantly be changing with the addition of new commands. I intend to implement somewhere from 8-12 commands, after which I will modify the DFA file to work with a table-based implementation. However, for the time being it is switch-case based.

Within the DFA file is two constant ArrayLists, that keep track of options for a command, if necessary, or parameters for a command if necessary. The idea is that the DFA can process input for some user input such as "mkdir 'C:/Users/User/newdirectory'" or "whoami" and return some result. The program will process the input separated by spaces and make judgment of what to classify the input based on the input and current state. With that being said, the general structure is: <command>, <command><option>, <command><option><param>,<command><param>, or <command><param>*. Where a command is the name of the command, an option is specified with a "-", and a parameter being any a-z character or number. Upon completion of being parsed, if the program is in an accepting state it will send the necessary ArrayList to the right object, and return some String result to the UI. In short, the general structure of this application is: A user-interface which interacts with the DFA file which then interacts with the appropriate command files and returns an output to the GUI.

Requirements

The program is relatively simple and hasn't been intensive with my usage of it. Any modern general-purpose computer will suffice.

Literature Survey

Similar systems that exist of course include bash shell implementations like GNOME, there's also Window's PowerShell too. I learned from our textbook that many CLI's use a DFA to some degree to parse commands. Afterwards I looked up how bash parses commands and it is quite different and tailored to the Linux environment unlike my terminal application so far.

User Manual

Since the application is still divided into several components it cannot be used at this time. Upon completion I will attach a file including all of the ways to use the commands. However thus far I have whoami which returns the current user, if sent with the "-help" option, it will tell the intention of the program. I also have the mkdir command which makes a directory and takes a parameter for a directory, and will either create it or not depending on if the path provided is valid. So valid commands are "whoami", "whoami -help" and "mkdir (some path)".

Conclusion

In short, this is my progress on the project thus far. I have worked on outlining a simple structure for how the application should work coinciding with various pieces such as UI, the DFA, and my implementations of the commands. I will continue to work on the project and implement more commands, so I can finalize the DFA table. Within this directory is a pdf containing the DFA diagram so far. It will be changed later to include the other commands that will be made in the upcoming weeks.