CS 100 Assignment 1: Design

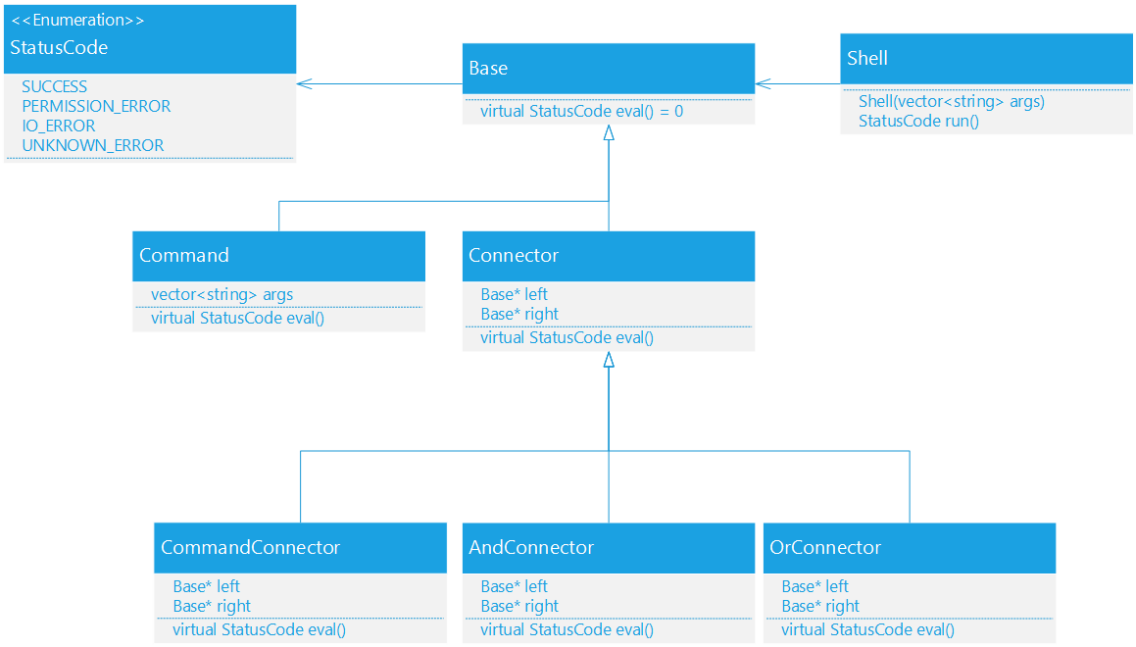
William Shiao, Siddhanth Sharma

April 26, 2017

1 Introduction

Our design is based around the Composite Pattern. Our RShell parses expressions into Commands and Connectors, which are then evaluated to generate the desired output.

2 UML Diagram



3 Classes/Class Groups

3.1 Shell

The `Shell` class is in charge of parsing user input and displaying the outputs. Its constructor also accepts arguments that can be used to change various settings.

3.2 StatusCode

In the case of errors, we included the `StatusCode` enum so that the `Base::eval` and `Shell::run` functions can return the result of commands, which can then be use to display a more detailed error message to the user.

3.3 Base

`Base` is an abstract base class that is the parent of all classes except for `StatusCode` and `Shell`.

3.4 Command

A `Command` is any user input that is meant to execute a program. It has a member variable, `args`, that stores any arguments to be passed into the program.

3.5 Connector

The `Connector` class is used for operations that connect two user inputs. It stores the left and right sides of the connector as `Base` objects, which are then evaluated to generate the output.

3.5.1 CommandConnector

The `CommandConnector` (;) class allows the user to write multiple commands on a single line.

3.5.2 AndConnector

The `AndConnector` (&&) class allows the user to execute two commands, where the second command executes only after the first one succeeds.

3.5.3 OrConnector

The `OrConnector` (||) class allows the user to execute two commands, where the second command executes only after the first one fails.

4 Coding Strategy

We will collaborate via GitHub by working on separate branches and merging our changes after testing to assure proper functionality. We will also agree on a code formatting standard and use Kanban to manage workflow, with occasional physical meetings to reinforce understanding.

5 Roadblocks

Roadblocks that will most likely occur include:

- Working at different speeds
- Conflicting views on the progress of the project
- Scheduling conflicts that could lead to unexpected errors regarding workflow
- Miscommunication and/or misunderstanding during development

The majority of potential roadblocks can be solved with proper work ethics and communication. As long as we manage our workflow efficiently (most likely with the use of a Kanban board) and work around potential conflicts with our work, these roadblocks should be manageable.