1 Описание архитектуры Bash

1.1 Bash

Bash - класс, который содержит всю реализацию командной строки. У него есть единственный метод Run, который запускает интерпритатор командной строки и завершается после прихода команды exit.

1.2 CommandsHandler

CommandsHandler - класс, обрабатывающий каждую поступающую команду. У него есть основной метод - Handle, принимающий строку, введенную пользователем. Он делает предобработку строкового представления команды с помощью Preprocessor'а (только для второй фазы), затем строит по строке ICommand и отдает ее на исполнение в Executor. Также у него есть метод HasExit, который вызывается из Bash после обработки каждой команды. В зависимости от возвращаемого им значения, интепретатор будет завершаться или продолжать работу.

1.2.1 State

State - класс, который хранит актуальные значения переменных окружения. У данного класса имеется 3 метода:

- UpdateVariable - обновить/добавить значение переменной.
- GetVariable - получить значение переменной по имени.
- GetEnv - получить значение всех переменных окружения для дальшейшей передачи в запускаемую программу.

1.3 Preprocessor

Preprocessor - класс, делающий подстановку переменных окружения, которые хранятся в State, в текст команды. Этот класс необходим только для второй фазы. У него есть один публичный метод DoPreprocessing, который принимает строковое представление команды и возвращает ее же, но с подставленными значениями переменных окружения.

1.4 Parser

Parser - класс, достает из исходной строки с командой, класс ICommand, который далее будет запускаться. Класс хранит в себе CommandCreator. У класса есть один метод - Parse, принимающий строку. В данном методе сначала создается Tokenizer для строки. С помощью него строка разбивается на токены, а затем полученная последовательность токенов отдается в CommandCreator, который уже создает команду ICommand.

1.4.1 Tokenizer

Tokenizer - класс, который разбивает строковое представление команды на токены. Этот класс в конструкторе принимает строковый поток и вычитывает из него токены по одному при помощи методов HasToken и GetToken.

1.4.2 CommandCreator

CommandCreator - класс, который строит по токенам команду ICommand. У него есть метод Create, принимающий последовательность токенов. По ней он определяет, что это за команда (например, является ли это программой с аргументам). В зависимости от этого он создает наследника ICommand и возвращает его в качестве результата.

ICommand ICommand - интерфейс, представляющий разные команды (например, команда установки переменной окружения или команда запуска программы с аргументами).

SimpleCommand SimpleCommand - обычная команда без пайпов (Пример: prog arg1 arg2 arg3). Хранит в себе название программы и ее аргументы.

CommandWithPipes CommandWithPipes - команда, состоящая из двух или более SimpleCommand, которые соединены с помощью пайпов (Пример: prog1 arg1 | prog2 arg2). Класс необходим для второй фазы.

EnvironmentUpdate EnvironmentUpdate - команда, содержащая изменение переменной окружения. Хранит в себе название переменной окружения и значение, которую в нее хотят записать.

1.5 Executor

Executor - класс, занимающийся выполнением ICommand. В конструкторе он принимает общий State с переменными окружения. У него есть публичный метод - Execute, принимающий ICommand. В зависимости от типа команды осуществляет ее выполнение. Для EnvironmentUpdate делает вызов соответствующего метода у State. Для SimpleCommand и CommandWithPipes он создает потоки для программ с помощью StreamBuilder'а. Затем с помощью ProgramResolver определяет, чем является каждая из программ в команде, и запускает их.

1.5.1 StreamBuilder

StreamBuilder - класс, который занимается созданием и перенаправлением потоков ввода и вывода между программами внутри одной комманды. У него есть метод CreatePrograms, который принимает ICommand и возвращает вектор из Program.

Program Program - класс, хранящий токены, относящиеся к данной программе, вместе с ее потоками ввода и выводами.

1.5.2 ProgramResolver

ProgramResolver - класс, который создает программу по токену с ее названием. У него есть метод ResolveProgram, принимающий Program и возвращающий IProgram.

IProgram IProgram - интерфейс для программы, готовой к исполнению.

UserProgram UserProgram - класс, представляющий любую пользовательскую программу.

BuiltinProgram BuiltinProgram - интерфейс, представляющий встроенную в Bash программу (например, cat). У него есть набор наследников - Echo, Cat, Wc и Pwd. В наследниках реализуются соответствующие команды.

2 Поддерживаемые сценарии работы

```
$ prog arg1 arg2 arg3
$ prog1 arg11 arg12 | prog2 arg21 arg22
$ A=some
$ echo $A
$ cat filename
$ pwd
$ wc
$ exit
$ x="ex"
$ y="it"
$ $x$y
$ prog $A arg2
$ exit | echo1
$ prog "$A kdkdk" arg2
$ prog '$A' arg2
$ prog1 "kfdfkddkf | fdsklsfdklsf " | prog2 arg21 arg22
$ pwd|wc
```