Техническая документация

Интерпретатор командной строки

Лямин Владимир, Ершов Владислав, Фомина Виктория

Октябрь 2021

Содержание

1 Общее об архитектуре CLI **2**

2 Компонент Parser **2**
2.1 Класс ParsedModel 3
2.2 Класс Parser 4

3 Компонент Executor **4**

4 Компонент CLI **5**

5 Компонент Variables **5**

6 Компонент Commands **6**

1 Общее об архитектуре CLI

Диаграммы компонентов и классов[1] описывают интерпретатор командной строки. Спроектированный интерпретатор состоит из пяти компонентов: Parser, Executor, CLI, Variables и Commands (рис. 1).

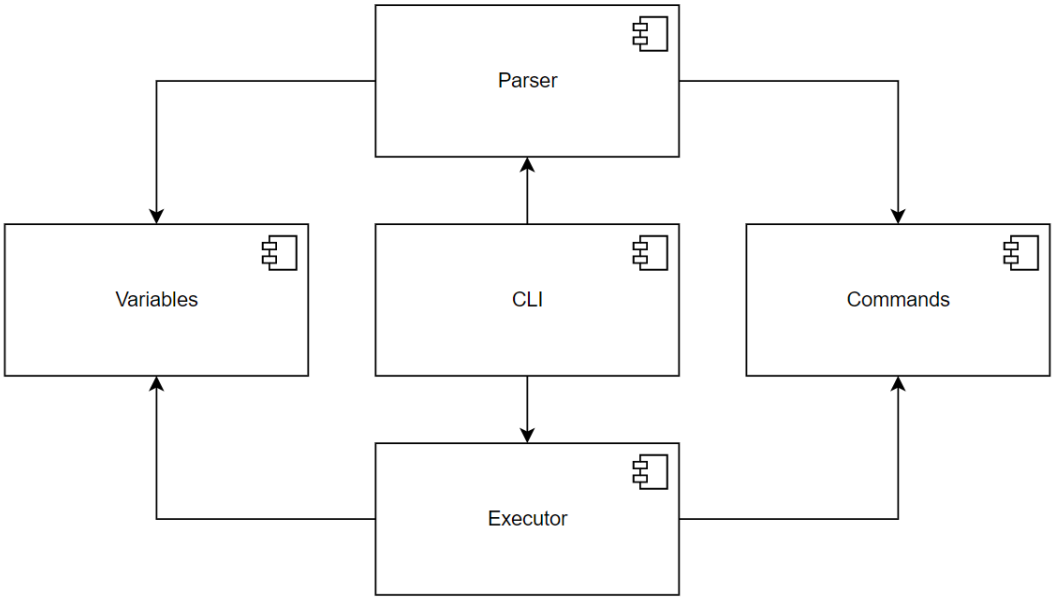

Рис. 1: Диаграмма компонентов

2 Компонент Parser

Задача компонента Parser (рис. 2) заключается в синтаксическом разборе входных данных и построению по ним модели, которая будет использована компонентом CLI для выполнения действий, задаваемых входными данными.

Два основных класса, из которых состоит компонент Parser:

- класс **Parser**, который занимается синтаксическим разбором строки, поступившей на вход интерпретатору;
- класс **ParsedModel**, который описывает модель возвращаемых из Parser'а данных.

[1] https://github.com/victoriafomina/Software-Design/tree/main/CLI

2.1 Класс ParsedModel

Класс ParsedModel содержит метод, позволяющий получить список объектов классов, реализующих интерфейс IAction (интерфейс обобщает два класса Command и Assignment) и задающих команду (в том числе и цепочку команд – pipeline (прим. присвоения не могут быть внутри pipeline)) или операцию присвоения значения переменной.

Класс Command необходим для хранения информации о команде (CommandType – свойство класса компонента Commands) и списка аргументов команды.

Класс Assignment используется для хранения информации о переменной и значении, которое ей необходимо присвоить.

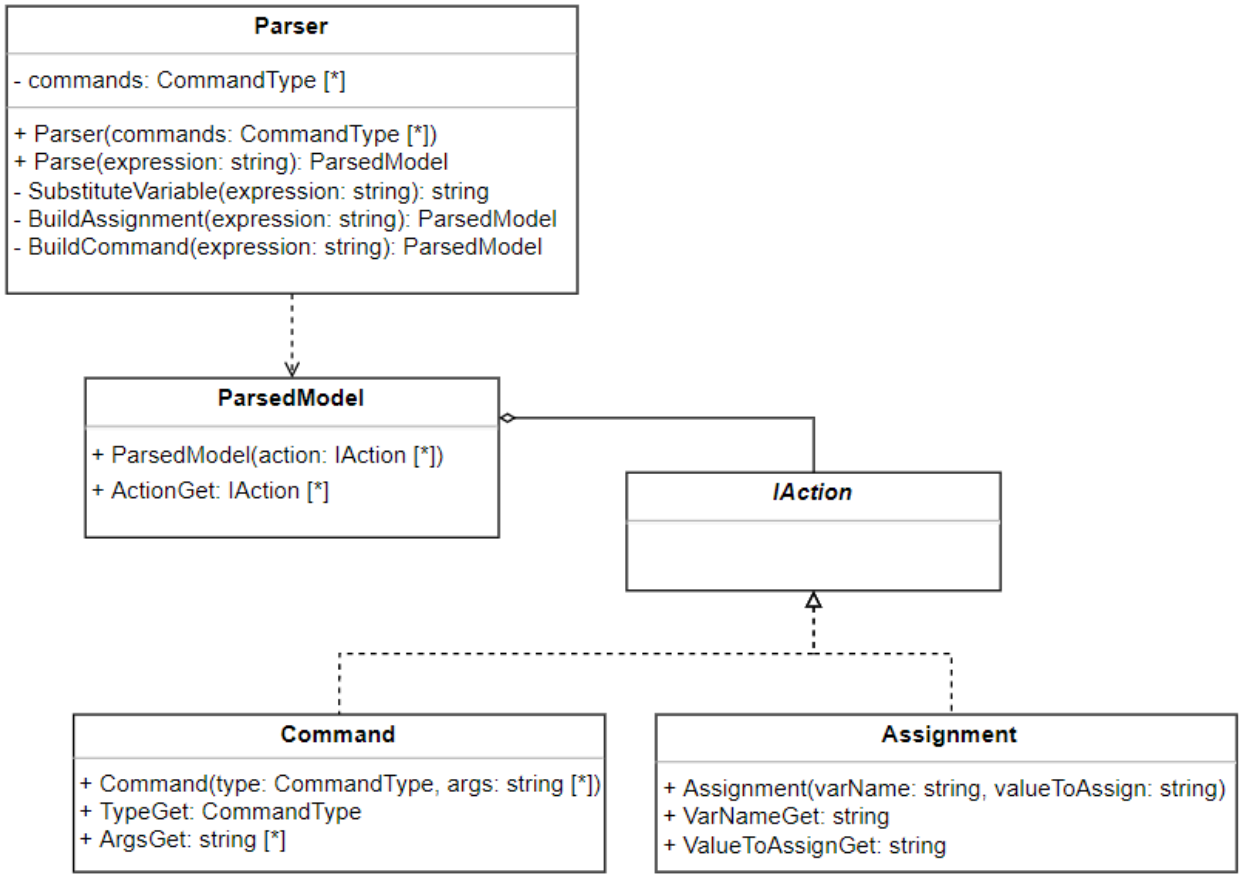

Рис. 2: Диаграмма классов компонента Parser

2.2 Класс Parser

Метод Parse класса Parser ответственен за синтаксический разбор входной строки. При вызове этот метод вызывает метод SubstituteVariable.

SubstituteVariable заменяет переменные, стоящие перед символом “$” на их значения (случаи с одинарными кавычками обрабатываются, двойные кавычки позволяют использовать пробелы в названиях переменных).

SubstituteVariable выполняет следующие действия:

1. заменяет символ “$” и названия переменных их значениями в случае если они не находятся внутри одинарных кавычек;
2. если после подстановки значений переменных появилась одна кавычка (одинарная или двойная), то считаем ее незначащим символом;
3. если после подстановки значения переменных появилось несколько кавычек, тогда берется первая в строке кавычка, находится первая парная для нее кавычка, а для оставшейся части строки выполняются предыдущие пункты, начиная с п. 1.

Далее по полученной из метода SubstituteVariable строке метод Parse определяет тип действия (команда или присвоение значения переменной) и запускает соответствующие методы BuildAssignment и BuildCommand (этот метод в том числе обрабатывает случаи с пайплайнами), которые возвращают объект класса ParsedModel, соответствующий определенному входной строкой действию.

3 Компонент Executor

Задача компонента Executor (рис. 3) заключается в отправке на исполнение действий посредством исполнения метода Execute, который в зависимости от того команда это (в т.ч. pipeline) или присвоение значения переменной запускает метод ExecuteCommand или метод ExecuteAssignment соответственно.

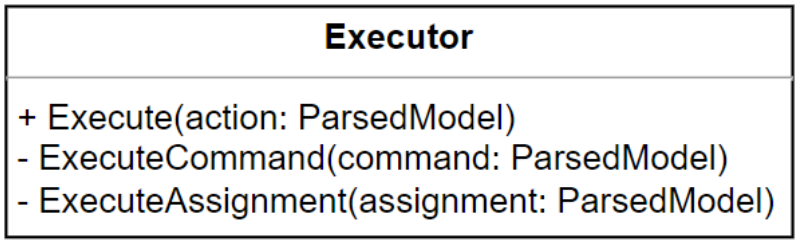

Рис. 3: Диаграмма классов компонента Executor

4 Компонент CLI

Задача компонента CLI (рис. 4) заключается в построчном считывании и обработке символов.

Компонент CLI состоит из класса CLI. Класс CLI включает в себя следующие методы:

- метод CallExternalProgram, вызывающий внешнюю программу;
- метод Run, выполняющий следующие действия:
  1. запуск интерпретатора командной строки;
  2. построчное считывание последовательности входных символов и отправка их в компонент Parse для проведения синтаксического разбора;
  3. получение объекта класса ParsedModel из компонента Parse и выполнении следующих действий в зависимости от контекста:
     - ParsedModel описывает команду (в т.ч. pipeline) или присвоение значения переменной – вызывается метод Execute класса Executor;
     - метод Parse выкинул исключение (встречена непонятая интерпретатору последовательность символов) – вызывается метод CallExternalProgram (поиск программы на исполнение осуществляется в директориях на которые указывают переменные среды).

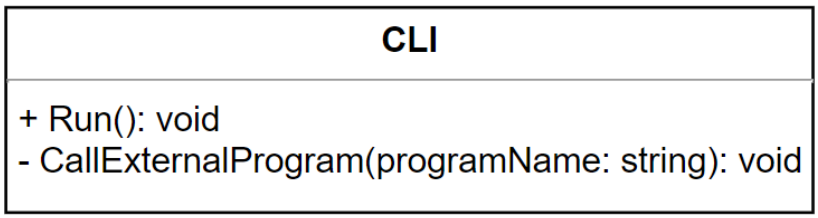

Рис. 4: Диаграмма классов компонента CLI

5 Компонент Variables

Задача компонента Variables (рис. 5) заключается в хранении, добавлении и обновлении значений переменных.

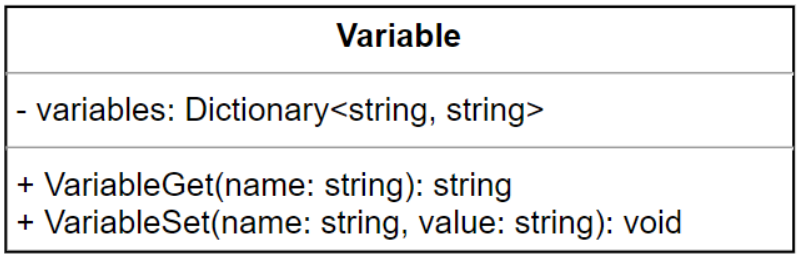

Рис. 5: Диаграмма классов компонента Variables

6 Компонент Commands

Задача компонента Commands (рис. 6) заключается в исполнении команд. Этот компонент также дает возможность получить информацию обо всех поддерживаемых интерпретатором командах.

Компонент Commands состоит из следующих классов:

- класса CommandType, хранящего информацию о команде (название, количество аргументов, тип возвращаемого значения);
- класса Commands, дающего информацию обо всех командах, поддерживаемых интерпретатором;
- классов PWDCommand, EchoCommand, ExitCommand и т.д. – наследников интерфейса ICommand;
- интерфейса ICommand требующего, чтобы класс команды, унаследованной от него, содержал метод Execute, принимающий и возвращающий поток. Также наследники интерфейса ICommand будут содержать поток ошибок.

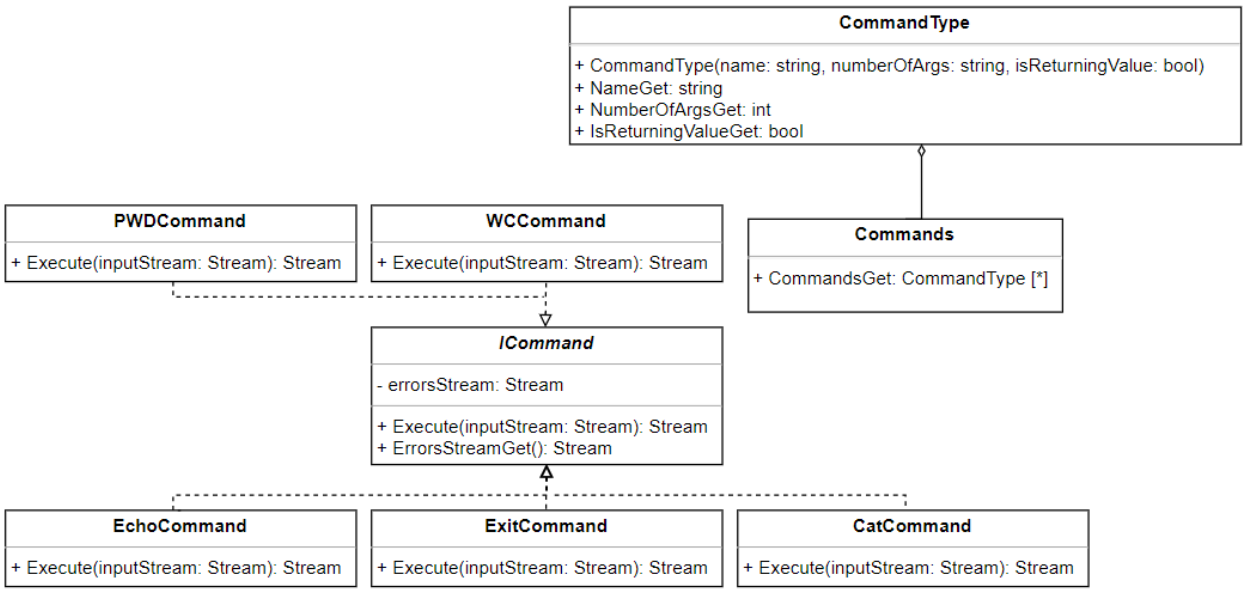

Рис. 6: Диаграмма классов компонента Commands