CLI Interpreter

Общая архитектура

Интерпретатор состоит из трех основных слоев: Core, Client, Infrastructure.

1. Client отвечает за чтение команд, вывод служебной информации и вывод ошибок.
2. Core содержит абстракции, всего приложения и задает поведение всему приложению. В этом компоненте мы задаем контракт общения с Client и Infrastructure, которому они должны следовать.
3. Infrastructure ответственен за хранение данных, необходимых для работы интерпретатора и команд. Таким образом, Core часть приложения обращается к Infrastructure для получения и изменения данных окружающей среды, Client читает данные об окружении из Infrastructure.

Реализация Client

Взаимодействие с интерпретатором производится через Client. Точка входа в интерпретатор класс Interpreter. В Interpreter вызывается метод Start который запускает процесс чтения ввода. Введенные данные кладутся в Buffer. Так же присутствует Task, которые читает из Buffer, если там лежит что-то, вызывается метод Process, который начинает выполнение. Он обращается к Parser.

Parser – класс, отвечающий за анализ входной строки.
Parse – метод класса Parser, который анализирует входную строку и возвращает список команд в виде списка объектов CommandDTO.
FormatInput – метод класса Parser, который определяет, есть ли в переданной строке подстроки, заключенные в двойные кавычки. Если есть, подстроки, начинающиеся со знака ‘$’, заменяются на соответствующие переменные среды. Для данных целей класс Parser содержит объект класса, реализующего интерфейс с IEnvironmentRepository.

Реализация Core

CommandDTO – вспомогательная структура, состоящая из полей, необходимых для дальнейшего построения дерева программы..
После работы парсера список CommandDTO передается в класс CommandService.

CommandService – класс, отвечающий за создание и исполнение команд. В конструкторе класса из списка объектов класса CommandDTO, с помощью приватного метода CreateTree создается древовидная структура из объектов INode. Затем команды исполняются вызовом метода Execute в корне дерева через приватный метод ExecuteTree.

INode – интерфейс, с помощью которого представляется узел в дереве. Производные классы должны реализовать контракт в виде метода Execute, принимающий структуру Response, содержащую сущности, отвечающие за потоки стандартного ввода, вывода и ошибок.

Command – абстрактный класс, реализующий интерфейс INode. Данный класс является базовым для встроенных и внешних команд интерпретатора. В их число входит команда, отвечающая за присвоение переменных среды.

Operation – абстрактный класс, реализующий интерфейс INode. Данный класс является базовым для операции пайпа (“|”).

С помощью данных абстрактных классов и интерфейса можно расширять набор команд и операций над командами.

Класс Command содержит поле, реализующее интерфейс IEnvironmentRepository. Данный интерфейс задает контракт по обращению команд к инфраструктуре.

Реализация Infrastructure

Environment – класс “одиночка”, в котором содержится потокобезопасный словарь, хранящий переменные среды. Также данный класс реализует методы по добавлению, изменению и получению переменных среды.