**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №5**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: Управление, разделение на уровни абстракции.

|  |  |  |
| --- | --- | --- |
| Студент гр. 0381 |  | Дзаппала Д. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2021

## Цель работы.

Реализовать управление игрой.

## Задание.

Необходимо организовать управление игрой (номинально через CLI). При управлении игрой с клавиатуры должна считываться нажатая клавиша, после чего происходит перемещение игрок или его взаимодействия с другими элементами поля.

Требования:

* Реализовать управление игрой. Считывание нажатий клавиш не должно происходить в классе игры, а должно происходить в отдельном наборе классов.
* Клавиши управления не должны жестко определяться в коде. Например, это можно определить в отдельном классе.
* Классы управления игрой не должны напрямую взаимодействовать с элементами игры (поле, клетки, элементы на клетках)
* Игру можно запустить и пройти.

## Выполнение работы.

Задание было выполнено благодаря реализованному паттерну «Команда» (Command).

Для начала, был написано класс интерфейс Command, с объявленным методом void execute(Character\*). Этот метод будет вызываться каждый раз, когда будет нажиматься та или иная кнопка. Класс Command наследуют классы LeftMove\_Command, RightMove\_Command, UpMove\_Command, DownMove\_Command, UseItemLHB\_Command, UseItemBHB\_Command, UseItemA\_Command. У каждого определен метод execute, который в зависимости от команды, вызывает для существа метод «сетта» направления движения. В случае использования предметов, вызывается метод UseItem.

Далее, был объявлен интерфейсный класс CommandHandler. В нем объявлен один виртуальный метод — command(), возвращающий указатель на команду — Command. Этот метод можно переопределять, используя графические библиотеки по выбору пользователя. В данном проекте используется графическая библиотека SFML.

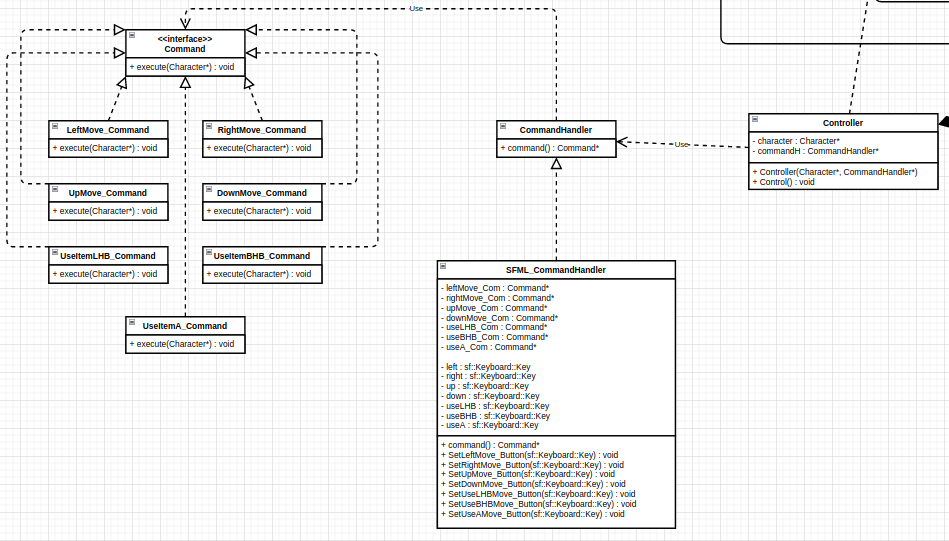
Именно поэтому, был реализован класс SFML\_CommandHandler, который наследует CommandHandler. Класс содержит указатели на все виды команд, а также поля «ключей» перечисления SFML — sf::Keyboard::Key. Нужны они, в случае, если пользователь захочет переопределить кнопки игры. Специально для этого, были написаны сеттеры под эти ключи-кнопки.

Был переделан класс Controller. Конструктор объекта принимает указатель на Character и CommandHandler для полей класса. Есть метод control(), в котором вызывается метод command() у указателя CommandHandler, и, если метод вернул указатель на команду, то вызывается метод execute данной команды, иначе ничего не происходит.

## Выводы.

Был реализован паттерн Команда, благодаря которому происходит управление игрой.

# Приложение А UML диаграмма классов



# Приложение Б Тестирование

Если результаты тестирования велики (больше 1 страницы), то их выносят в приложение.

Процесс тестирования можно представить в виде таблицы, например:

Таблица Б.1 - Примеры тестовых случаев

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| ... |  |  |  |

Обратите внимание, что в нумерации таблицы в приложении обязательно должен быть в качестве префикса номер самого приложения: А.