МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Объектно-ориентированное программирование»

Тема: уровни абстракции, управление игроком

Студент гр. 1384 Прошичев А.В.

Преподаватель Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Приобрести навыки создания уровней абстракции. Сделать управление игрока более гибким за счёт возможности изменения клавиш.

Задание.

Реализовать набор классов отвечающих за считывание команд пользователя, обрабатывающих их и изменяющих состояния программы (начать новую игру, завершить игру, сохраниться, управление игроком, и.т.д.). Команды/клавиши определяющие управление должны считываться из файла.

Требования:

- Реализован класс/набор классов обрабатывающие команды
- Управление задается из файла
- Реализованные классы позволяют добавить новый способ ввода команд без изменения существующего кода (например, получать команды из файла или по сети).
- Из метода считывающего команду не должно быть “прямого” управления игроком

Примечания:

- Для реализации управления можно использовать цепочку обязанностей, команду, посредника, декоратор, мост, фасад

Выполнение работы.

Изначально в прошлых лабораторных работах для обработки команд использовался всего один класс. В данной же лабораторной работе он был разбит на множество классов, между каждый из которых были распределены обязанности.

Класс, который правильно комбинирует работу методов других классов, отвечающих за считывание и вывод информации, называется InputCenter.

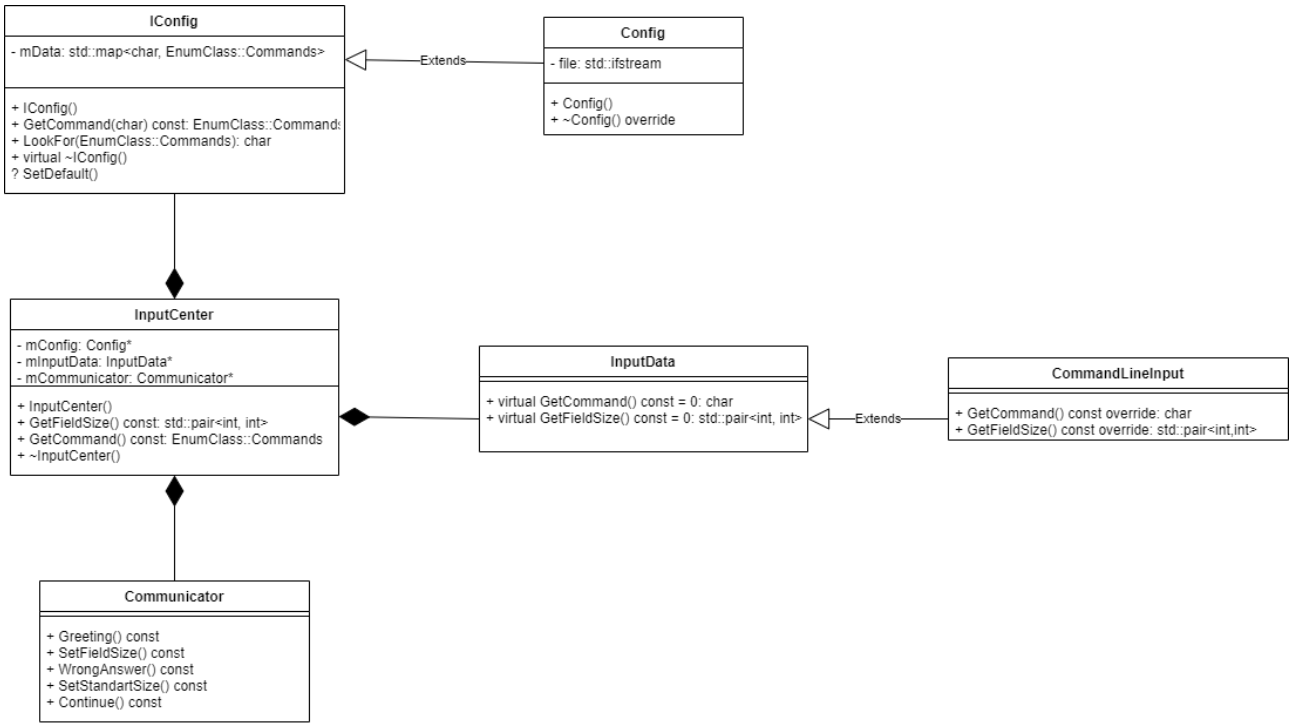
Данный класс связан с Медиатором и получает задания, который должен выполнить, чтобы игра получала обратную связь от игрока. Например, метод GetFieldSize() обращается к классу Communicator для вывода информационного сообщения на экран пользователя, далее он получает данные от класса InputData, который отвечает за считывание команд из определённого потока. Чтобы понять полученный команды, метод обращается к классу Config, который хранит связь между командой-символом и командой-перечислением.

Класс Communicator – класс с набором методов, отвечающих за вывод информации для игрока. Ничего не возвращает и не считывает.

Класс InputData – интерфейс, реализация которого позволит считывать обратную связь от пользователя через определённый поток, привязанный к конкретной реализации данного интерфейса. Имеет два метода: метод для считывание символьной команды GetCommand() и метод для считывание пары чисел GetFieldSize()

Класс CommandLineInput является реализацией названного выше класса. Он отвечает за считывание данных из консоли.

Класс Config умеет обрабатывать файл config.txt, хранящий параметры, который определяют клавиши для взаимодействия с игрой. При создании экземпляра этого класса открывается соотвествующий текстовый файл, из него считываются команды и записываются в словарь по ключу - символьной команде и по значению – команде – перечислению. Далее с данным классом взаимодействие происходит через обращению к методам, которые по-разному обрабатывают словарь, например, находят ключ по значению GetCommand() или наоборот – LookFor().



Тестирование.

```
Do you want to enter a size of map yourself? [Y/N]
Y
```

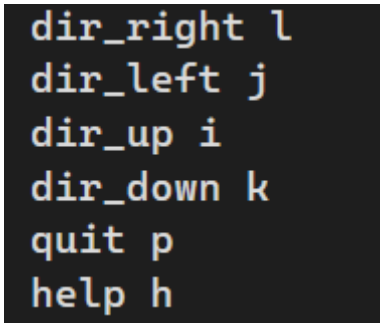
Изображение №1. Выбор размера поля.

```
The size of the map must be in range from 3 to 20
If you enter an incorrect output, the values will move to the rigth range
Enter the size of the map, separating it with a space.
Format of the enter: Width Height
20 30
```

Изображение №2. Пользовательские настройки поля.

```
q
Incorrect command! Please, check the correct format of the enter!
```

Изображение №3. Использование несуществующих команд.



Изображение №4. Файл Config.txt

Выводы.

Приобрёл навыки создания уровней абстракции. Сделал управление игрока более гибким за счёт возможности изменения клавиш.