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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Объектно-ориентированное программирование»

Тема: Интерфейсы, динамический полиморфизм

Студент гр. 1304 _____ Павлов Д.Р.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2022

Цель работы.

Реализовать систему событий. Событие - сущность, которая срабатывает при взаимодействии с игроком. Должен быть разработан класс интерфейс общий для всех событий, поддерживающий взаимодействие с игроком. Необходимо создать несколько групп разных событий реализуя абстрактный класс унаследованный от интерфейса события (например, враг, который проверяет условие, будет ли воздействовать на игрока или нет; ловушка, которая безусловно воздействует на игрока; событие , которое меняет карту; и.т.д.). Для каждой группы реализовать конкретные события, которые по разному воздействуют на игрока (например, какое-то событие заставляет передвинуться игрока в определенную сторону, а другое меняет характеристики игрока). Также, необходимо предусмотреть событие “Победа/Выход”, которое срабатывает при соблюдении определенного набора условий.

Реализовать ситуацию проигрыша (например, потери всего здоровья игрока) и выигрыша игрока (добрался и активировал событие “Победа/Выход”)

Требования.

Разработан интерфейс события с необходимым описанием методов

Реализовано минимум 2 группы событий (2 абстрактных класса наследников

события)

Для каждой группы реализовано минимум 2 конкретных события (наследники от

группы события)

Реализовано минимум одно условное и безусловное событие (условное - проверяет выполнение условий, безусловное - не проверяет).

Реализовано минимум одно событие, которое меняет карту (меняет события на

клетках или открывает расположение выхода или делает какие-то клетки

проходимыми (на них необходимо добавить события) или не непроходимыми

Игрок в гарантированно имеет возможность дойти до выхода

Описание архитектурных решений и классов.

В данной лабораторной работе были ***описывается паттерн Abstract Factory***.

Описание классов:

1) ***Abstract_Event_Factory*** — Абстрактная фабрика. Это порождающий шаблон проектирования, предоставляет интерфейс для создания семейств взаимосвязанных или взаимозависимых объектов, не специфицируя их конкретных классов. Шаблон реализуется созданием абстрактного класса Factory, который представляет собой интерфейс для создания компонентов системы (например, для оконного интерфейса он может создавать окна и

кнопки). Затем пишутся классы, реализующие этот интерфейс. Имеет два пустых Виртуальных метода, возвращающих указатель на Event: create_event_first и create_event_second.

2) ***Event*** — Абстрактный Класс события, имеющий виртуальный метод do_something, который пустой.

Положительные Ивенты:

3) ***Positive_Event*** — Класс-Наследник от Абстрактной Фабрики. Перегружает методы последней на возврат new Heal и new XP.

4) ***Heal*** — Класс-Наследник от Ивента. Перегружает метод последней на прибавку Герою ХП (условие: только если ХП Героя <= 9).

5) ***XP*** — Класс-Наследник от Ивента. Перегружает метод последней на прибавку Герою Опыта.

Отрицательные Ивенты:

6) ***Negative_Event*** — Класс-Наследник от Абстрактной Фабрики. Перегружает методы последней на возврат new Enemy и new Teleport.

7) ***Enemy*** — Класс-Наследник от Ивента. Перегружает методы последней на -1 ХП (Если ХП == 0 — Герой умер).

8) ***Teleport*** — Класс-Наследник от Ивента. Перегружает методы последней на телепорт Героя в рандомную пустую клетку.

Глобальные Ивенты:

9) ***Global_Event*** — Класс-Наследник от Абстрактной Фабрики. Перегружает методы последней на возврат new User_Win и new Refresh_Events.

10) ***User_Win*** — Класс-Наследник от Ивента. Перегружает методы последней на Победу Героя.

11) ***Refresh_Events*** — Класс-Наследник от Ивента. Перегружает методы последней на создание новых объектов-событий.

Демонстрация работы программы и тестирование.

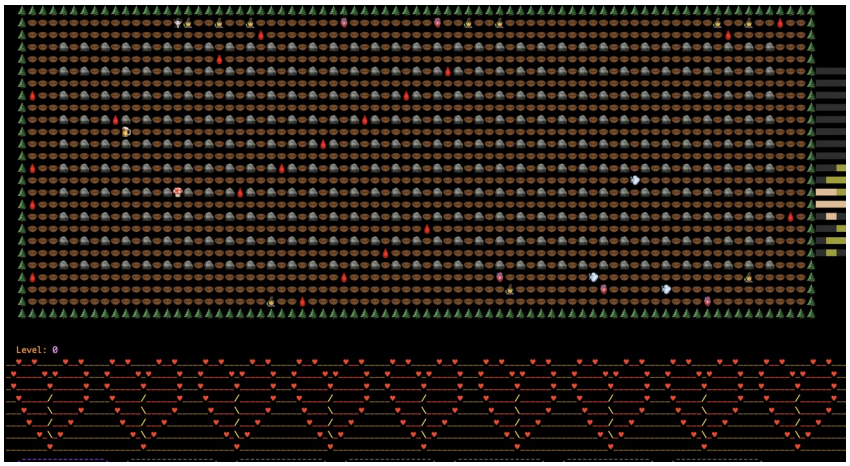


Рисунок 1 — До наступания на противника (ХП — 9)

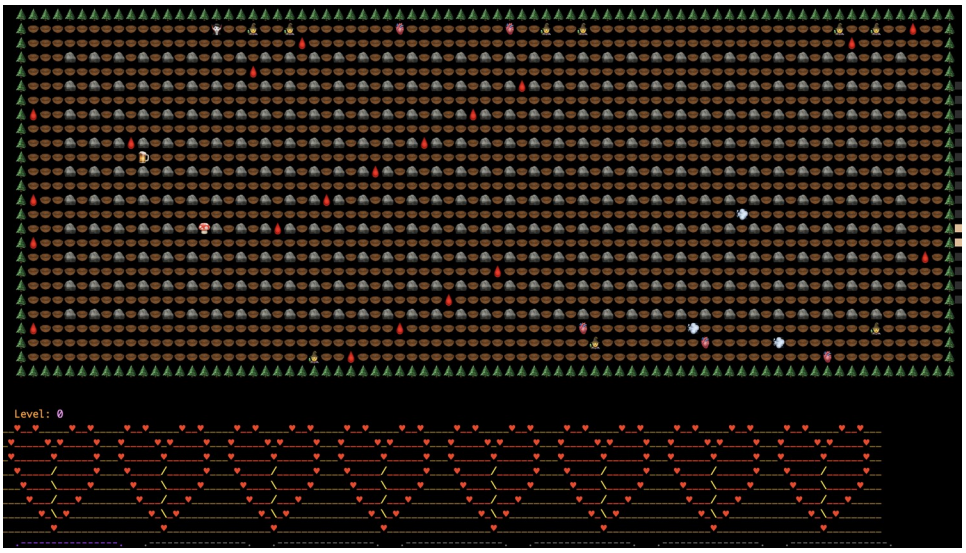


Рисунок 2 — После (ХП стало 8)

Из Рисунка 1 и 2 понятен принцип работы врага.

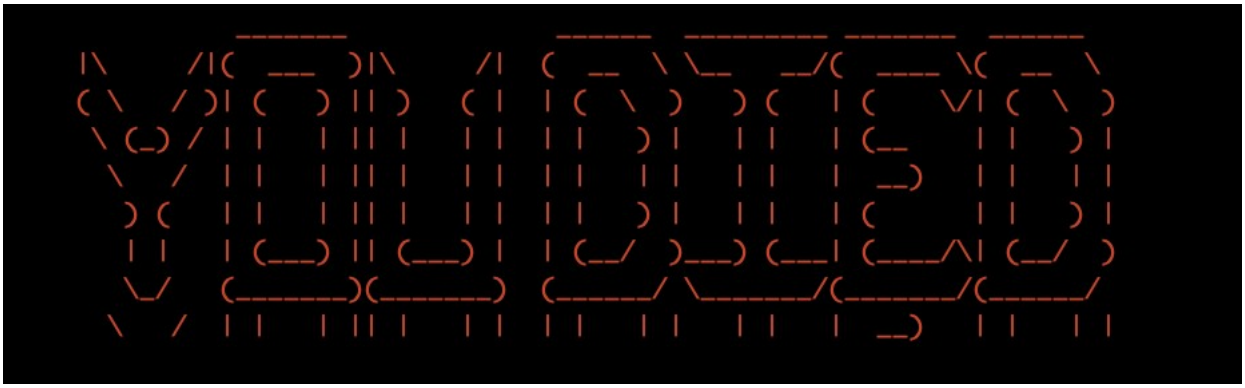


Рисунок 3 — Смерь Героя

Для победы Игрока необходимо убить всех врагов и достигнуть >=1 уровня. В качестве победной клетки была выбрана иконка «Пиво». (Рисунок 4)

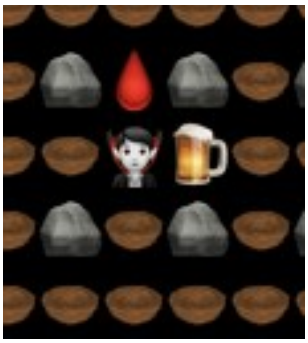


Рисунок 4 — Герой хочет Пиво, но не может выпить, поскольку за ним охотятся Враги.

Для того, чтобы собирать опыт и, как следствие, повысить уровень, Герою необходимо собирать «Кровь».

Так же на поле присутствуют иконки «Веторок» и «Человеческое сердце» - это Телепорт и +ХП. Так же есть Гриб, который добавляет на поле новые клетки событий (событие Refresh_Events). (Рисунок 5-6)

Рисунок 5 — Ветерок и ХП

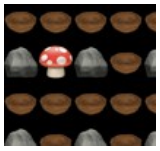
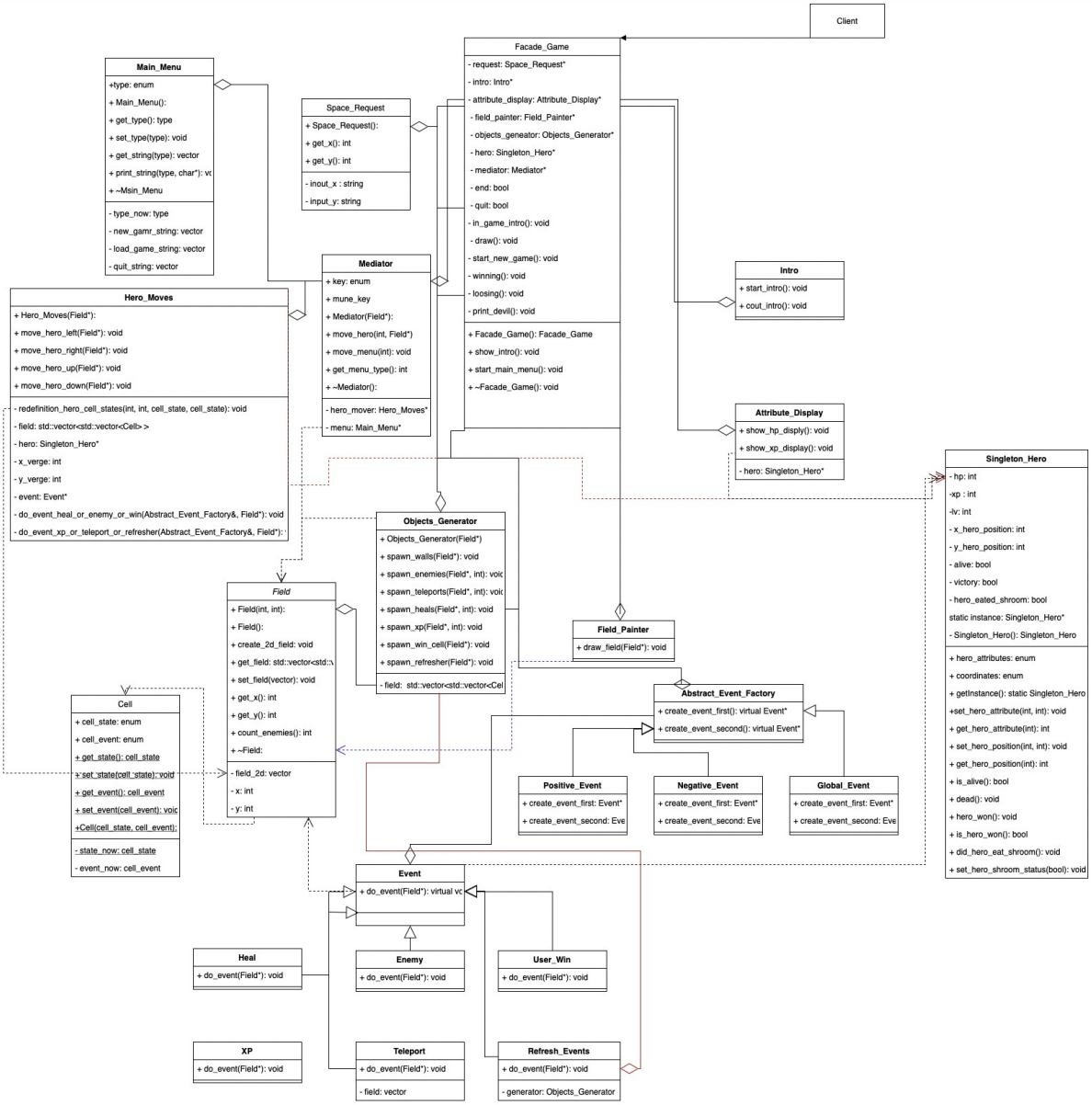


Рисунок 6 - ГРИБ

Если Герой все-таки убил всех врагов и достиг 1-го уровня, а после пошел за пивом, то его ждет победа. (Рисунок 7)

Рисунок 7 - Победа

UML диаграмма проекта



Вывод.

Реализована система событий и ее взаимодействие с игрой. В основе реализации системы событий лежит паттерн Абстрактная Фабрика.

Была изучена работа с классами на языке C++, паттерны проектирования, основы составления UML-диаграмм.