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: шаблонные классы, управление

Студент гр.0382 _____ Ильин Д.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2021

Цель работы.

Изучить применение шаблонов. Написать класс игры, параметризуемый правилами; данный класс изменяет состояние игры.

Задание.

Необходимо определить набор правил для игры в виде классов (например, какие задачи необходимо выполнить, чтобы он мог выйти с поля; какое кол-во врагов и вещей должно быть на поле, и.т.д.). Затем определить класс игры, которое параметризуется правилами. Класс игры должен быть прослойком между бизнес-логикой и командами управления, то есть непосредственное изменение состояния игрой должно проходить через этот класс.

Требование:

- Созданы шаблонные классы правил игры. В данном случае параметр шаблона должен определить конкретные значения в правилах.
- Создан шаблонный класс игры, который параметризуется конкретными правилами. Класс игры должен проводить управление врагами, передачей хода, передавать информацию куда переместить игрока, и.т.д.

Потенциальные паттерны проектирования, которые можно использовать:

- *Компоновщик (Composite) - выстраивание иерархии правил*
- *Фасад (Facade) - предоставления единого интерфейса игры для команд управления*
- *Цепочка обязанностей (Chain of Responsibility) - обработка поступающих команд управления*
- *Состояние (State) - отслеживание состояние хода / передача хода от игрока к врагам*
- *Посредник (Mediator) - организация взаимодействия элементов бизнес-логики*

Выполнение работы.

Чтобы наша игра параметризовалась правилами, были написаны шаблонные классы для них. Было решено написать всего 2 правила: правило, задающее строитель игрового поля и правило, определяющее количество врагов, которое может остаться при окончании игры.

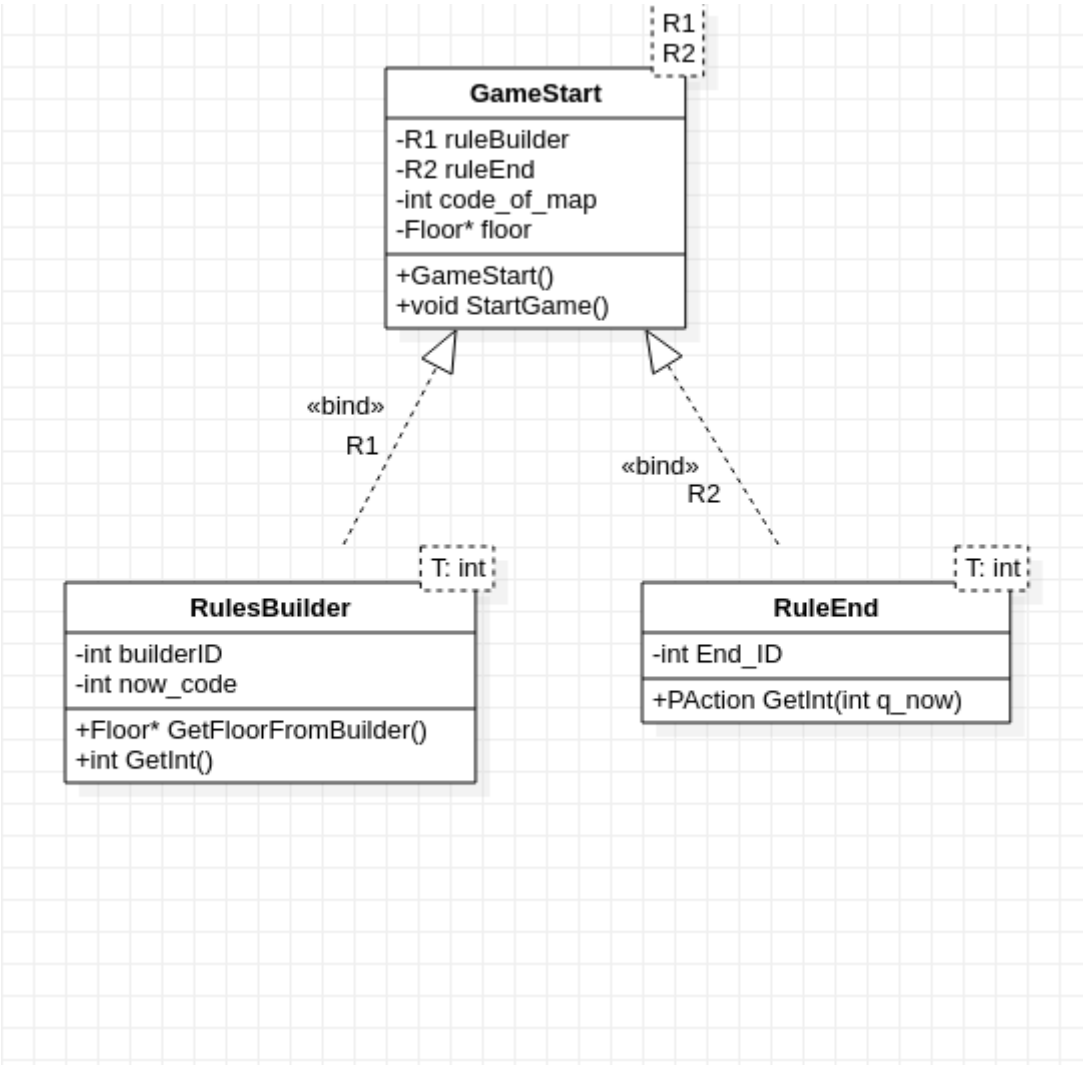
Первому из них соответствует шаблонный класс template<int T> class RulesBuilder. В нём создаётся игровое поле определённого формата, если T меньше нуля, то мы получаем псевдо-рандомно построенное поле, если равно нулю, то тестовое поле, если больше нуля, то получаем поле созданное в соответствии с T.

Второму из них соответствует шаблонный класс template<int T> class RuleEnd. При помощи него происходит проверка того, сколько осталось врагов на игровом поле и можно ли при таком количестве считать игру пройденной.

Собственно, данные правила должны параметризовать шаблонный класс игры, в котором еще и происходит управление изменением состояния игры. Для этого мы переопределяем класс Game. Делаем его шаблонным классом template<typename R1, typename R2> class Game. Предполагается, что при создании экземпляра данного класса первому параметру R1 будет соответствовать RuleBuilder<int>, второму параметру R2 будет соответствовать RuleEnd<int>. Для хранения правил, согласно которым происходят какие-либо действия, были также объявлены поля: R1 ruleBuilder (что задает, какого строителя выбрать), R2 ruleEnd (правило прохождения). Какие конкретные экземпляры правил там будут храниться – задается аргументами конструктора экземпляра шаблонного класса Game. В этом же классе реализован метод StartGame(). Он отвечает за ведение игры.

UML-диаграмма классов представлена на рис. 1.

Рисунок 1 – UML-диаграмма классов.



Выводы.

В ходе работы было изучено применение шаблонов; были написаны шаблонные классы правил, которые параметризуют написанный шаблонный класс игры.