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

ОТЧЕТ

по лабораторной работе №6

по дисциплине «Объектно-ориентированное программирование»

ТЕМА: Сериализация, исключения.

Студентка гр. 0382 _____ Чегодаева Е.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2021

Цель работы.

Изучить сериализацию и исключения.

Реализовать сохранение и загрузку игры в любой момент, создать набор исключений, которые срабатывают, если файл с сохранением некорректный.

Задание.

Сериализация - это сохранение в определенном виде состоянии программы с возможностью последующего его восстановления даже после закрытия программы. В рамках игры, это сохранения и загрузка игры.

Требования:

- Реализовать сохранения всех необходимых состояний игры в файл
- Реализовать загрузку файла сохранения и восстановления состояния игры
- Должны быть возможность сохранить и загрузить игру в любой момент
- При запуске игры должна быть возможность загрузить нужный файл
- Написать набор исключений, который срабатывают если файл с сохранением некорректный
- Исключения должны сохранять транзакционность. Если не удалось сделать загрузку, то программа должна находится в том состоянии, которое было до загрузки. То есть, состояние игры не должно загружаться частично

Потенциальные паттерны проектирования, которые можно использовать:

- *Снимок (Memento) - получение и восстановления состояния объектов при сохранении и загрузке*

Выполнение работы.

Class Game — класс игры. Был расширен добавлением новых методов: Save() и Load().

- Save() — метод для сохранения текущего состояния поля. Сохранение осуществляется путём записи в файл с названием "Save.txt" строк с информацией о поле, порядке объектов и их вида, местоположения игрока и его характеристики.

  Таким образом, после применения метода, в файле содержатся следующие строки в строгом порядке:

<Статус игры>
<Высота поля> <Ширина поля>
<Координата по Ox> <Координата по Oy> <Тип клетки> <Наличие объекта> <Тип объекта>
……..
<Координата по Ox> <Координата по Oy> <Тип клетки> <Наличие объекта> <Тип объекта>
<Координата героя по Ox> <Координата героя по Oy>
<Текущее HP героя> <Текущая броня героя> <Текущая атака героя>

(Где Статус игры = завершена/не завершена; Тип клетки: пустая, свободная и тд; Наличие объекта = есть ли на клетке монстр/предмет; Тип объекта представляет собой тип встретившегося объекта – при НЕ нулевом прошлом пункте-)

При сохранении записывается состояние всех клеток на поле.

- Load() — метод для запуска сохранённого состояния поля. Чтение осуществляется из того же файла "Save.txt" путём посимвольного считывания заданных строк. При считывании создаётся пустое игровое поле, по размерам сохранённого, и параметры каждой клетки заполняются информацией, получаемой из файла. В итоге “промежуточное” поле полностью отображает состояние игрового поля до последнего сохранения. Кроме того, аналогичным способом задаются характеристики игрока (через “промежуточного” TMP_User).

Далее, когда все данные получены - параметры “временных” объектов переносятся на действующее и тем самым позволяя пользователю продолжить сохранённую игру.

На каждом шаге считывания осуществляется проверка получаемых данных, и в случае их повреждения (или явного несоответствия ожидаемым) – вбрасывается собственное исключение определённого типа.

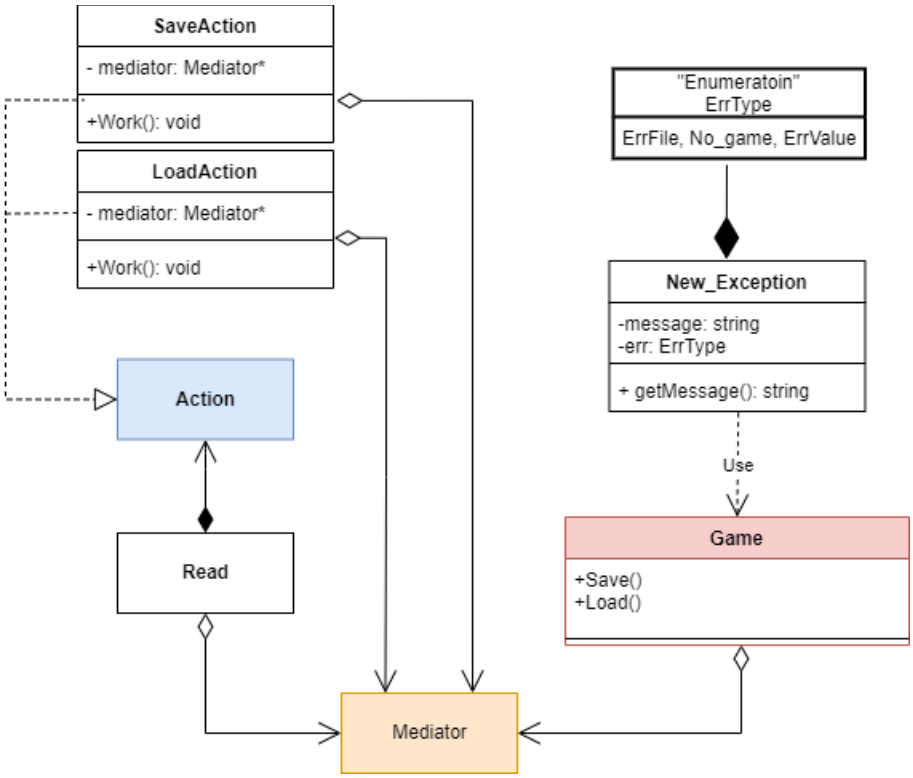
Class New_Exception — класс исключения. В качестве конструктора принимает тип встретившейся ошибки (типы ошибок описаны в ErrType) и далее присваивает полю message советующее сообщение. Данное сообщение будет возвращено пользователю в месте встречи ошибки. Внутри класса описаны 3 возможных исключения:

- ✓ ErrFile – ошибка при попытке получить доступ к файлу.
- ✓ No_game – игра была завершена (То есть проложить игру будет невозможно).
- ✓ ErrValue – один (или несколько) параметров, необходимых при считывании, содержит недопустимое значение.

Помимо этого, для возможности сохранения и запуска игры в любой момент (то есть ВО ВРЕМЯ ИГРЫ) был расширен класс Read(), добавлены новые классы SaveAction и LoadAction (подобные иным подклассам Action), что позволяет пользователю при вводе соответствующих клавиш обращаться к тому или иному методу.

Так же осуществляется корректный запуск игры – “Продолжить старую игру “, с сохранением логгирования и остальными возможностями (Рассмотрены 2 случая “запуска игры” – новая игра и игра, по сохранённым ранее данным).

UML – диаграмма:



Выводы.

Были изучены сериализация и исключения.

Реализовано сохранение всех необходимых состояний игры в файл и загрузка игры в любой момент, также создан набор исключений, которые срабатывают, если файл с сохранением некорректный. Представленные исключения сохраняют транзакционность, состояние игры прогружается полностью.