**Федеральное государственное бюджетное образовательное учреждение**
**высшего образования «Университет «Дубна»**
Институт системного анализа и управления
Кафедра распределенных информационно-вычислительных систем

# КУРСОВАЯ РАБОТА

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

**Тема:** Применение объектно-ориентированного программирования для разработки игры «Сапёр»

**Выполнил** студент группы: 1012

Трусов И. А.

(подпись студента)

**Руководитель:**

ст.преп. Ушанкова М.Ю.

(подпись преподавателя)

**Оценка:**

**Дата защиты:**

Дубна, 2024

# Оглавление

# Введение в работу

Современные игры требуют не мало ресурсов для запуска, чтобы в них поиграть и не на каждом устройстве возможно поиграть. Игры способствуют быстрому развитию логического мышления и внимательности. Поэтому была выбрана тема по созданию для развития навыков, доступной, а значит простой игры, которая не будет требовать выход в интернет. Актуальность выбранной темы обусловлена тем, что объектно-ориентированное программирование (ООП) является одним из наиболее популярных подходов к разработке программного обеспечения. Его применение позволяет создавать сложные и масштабируемые приложения, упрощает процесс разработки и поддержки кода. Одной из областей, где ООП может быть эффективно применено, является разработка компьютерных игр. В данной работе будет рассмотрена возможность использования ООП для создания игры «Сапёр».

Объектно-ориентированный подход позволяет структурировать код таким образом, чтобы он был более понятным и легко поддерживаемым. Это особенно важно при разработке сложных приложений, таких как компьютерные игры. В рамках данной работы будет рассмотрено, как использование ООП может улучшить процесс разработки игры «Сапёр»[5].

# Цели и задачи

Цель работы заключается в исследовании возможностей применения ООП для разработки игры «Сапёр» и создании прототипа игры с использованием данного подхода.

Для достижения поставленной цели необходимо решить следующие задачи:

- 1. Изучить основы объектно-ориентированного программирования.
- 2. Разработать структуру классов для игры «Сапёр».
- 3. Реализовать функционал игры с помощью созданных классов.
- 4. Провести тестирование и отладку программы.

Ожидаемый результат: Игра, включающая несколько уровней сложностей, протестированная с минимальным количеством незначительных ошибок.

# Формулировка задачи

Цель: Дать возможность пользователю убить время.

Исходные данные: Принципы ООП, правила *“Minesweeper”*[1].

Модельное представление: Игра сапёр, в начале которой создаётся поле из клеток. Игроку сообщается количество мин на поле. Цель игрока — открыть все клетки, не содержащие мин.

Ожидаемый результат: Игра, включающая несколько уровней сложностей, протестированная с минимальным количеством незначительных ошибок.

Критерии оценки результата: Увеличение времени проведенного в приложении, повышение частоты повторных посещений, увеличение уровня вовлеченности пользователя.

# Проектирование

- Создание нового проекта.
- Создание классов.
- Создание пользовательского интерфейса.
- Связка пользовательского интерфейса с элементами кода классов.

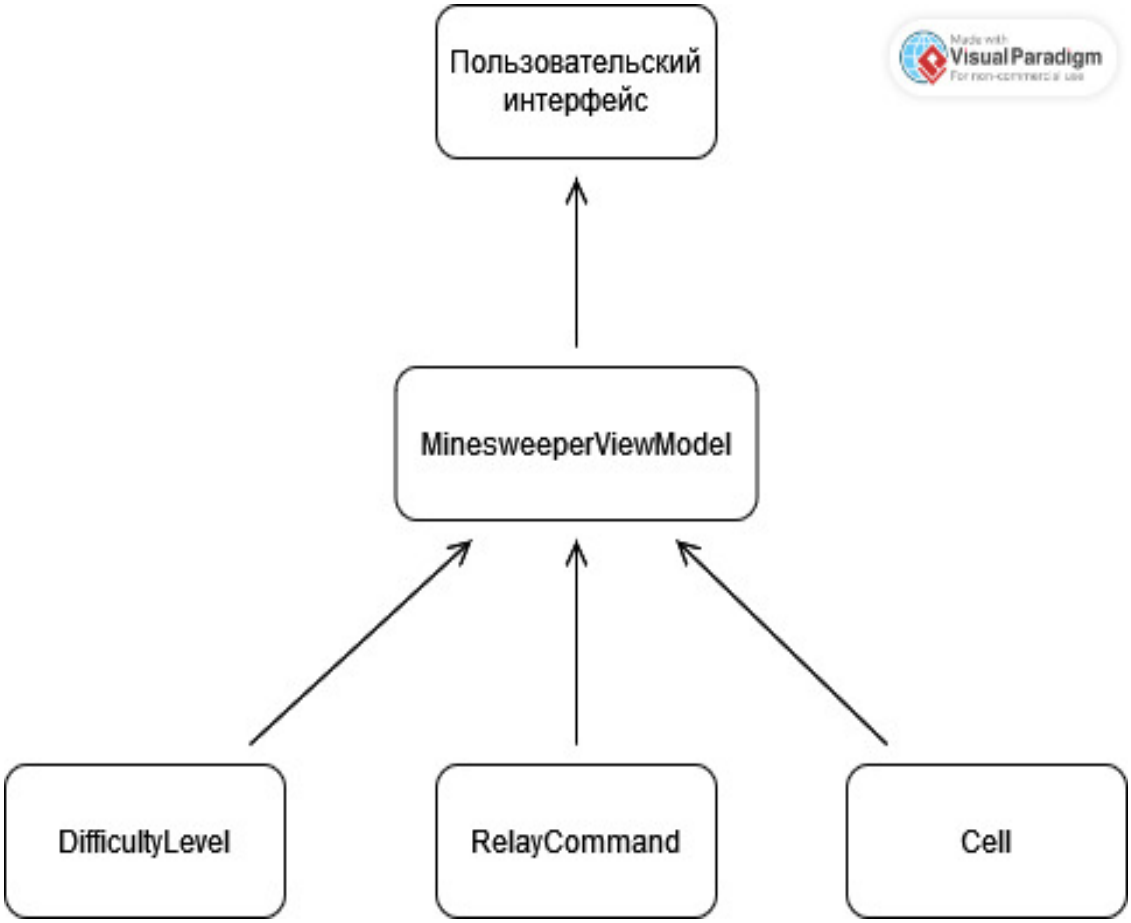


Рисунок 1. Структурой приложения.

Спроектировано четыре класса: модельное представление сапёра, ячейка, сложность уровня, исполнения команд. Класс модельное представление сапёра, связан с классами, ячейка, сложность уровня и исполнения команд, агрегацией. Класс модельное представление сапёра, должен предавать все возможные команды и изменения в пользовательский интерфейс. Для создания работы был выбран язык программирования *C#*. Для разработки было выбрано *Microsoft Visual Studio* версии 2022 года. Для разработки пользовательского интерфейса было выбрано оконное приложение *Windows Presentation Foundation* (*WPF*) с использованием фреймворка *.NET*.

# Разработка

С помощью модульной платформы для разработки *.NET* и использования платформы *Windows Presentation Foundation* (*WPF*) создан проект для разработки игры. Подключено две библиотеки *System.Windows.Input* и *System.ComponentModel*. Выбор *WPF* был, сделан в пользу текстовой разметки *XAML* и встроенных библиотек. *XAML* можно связывать код с объектами разметки при помощи связки (*binding*).

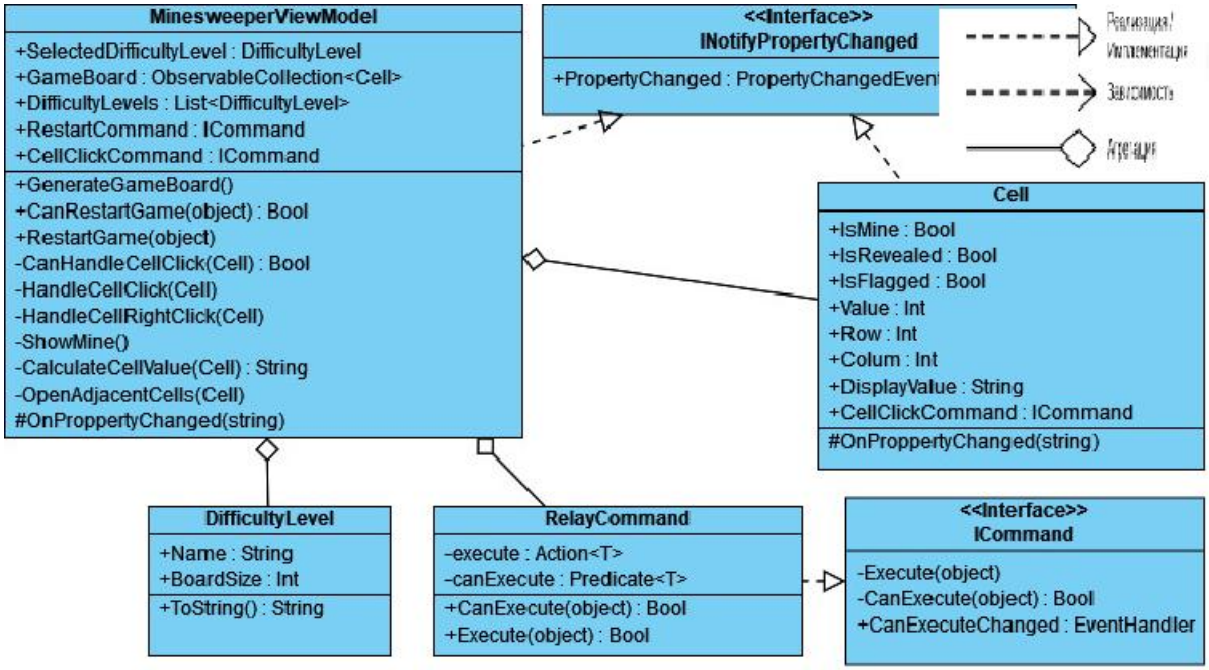


Рисунок 2. UML-диаграммы классов.

Разработано в проекте четыре класса: *RelayCommand, DifficultyLevel, Cell, MinesweeperViewModel*. Класс *RelayCommand* будет запоминать действия команд, наследует из библиотеки *System.Windows.Input* интерфейс *ICommand*, который определяет действия команд. Класс *DifficultyLevel* будет определять сложность и размер игрового поля. Класс *Cell* будет описывать каждую создавшуюся ячейку, наследует из библиотеки *System.ComponentModel* интерфейс *INotifyPropertyChanged* который сообщает об изменении значений свойства класса. Класс *MinesweeperViewModel* содержит в себе логику и данные взаимодействия с пользовательским интерфейсом, наследует из библиотеки *System.ComponentModel* интерфейс *INotifyPropertyChanged* который сообщает об изменении значений свойства (см. рис. 2).

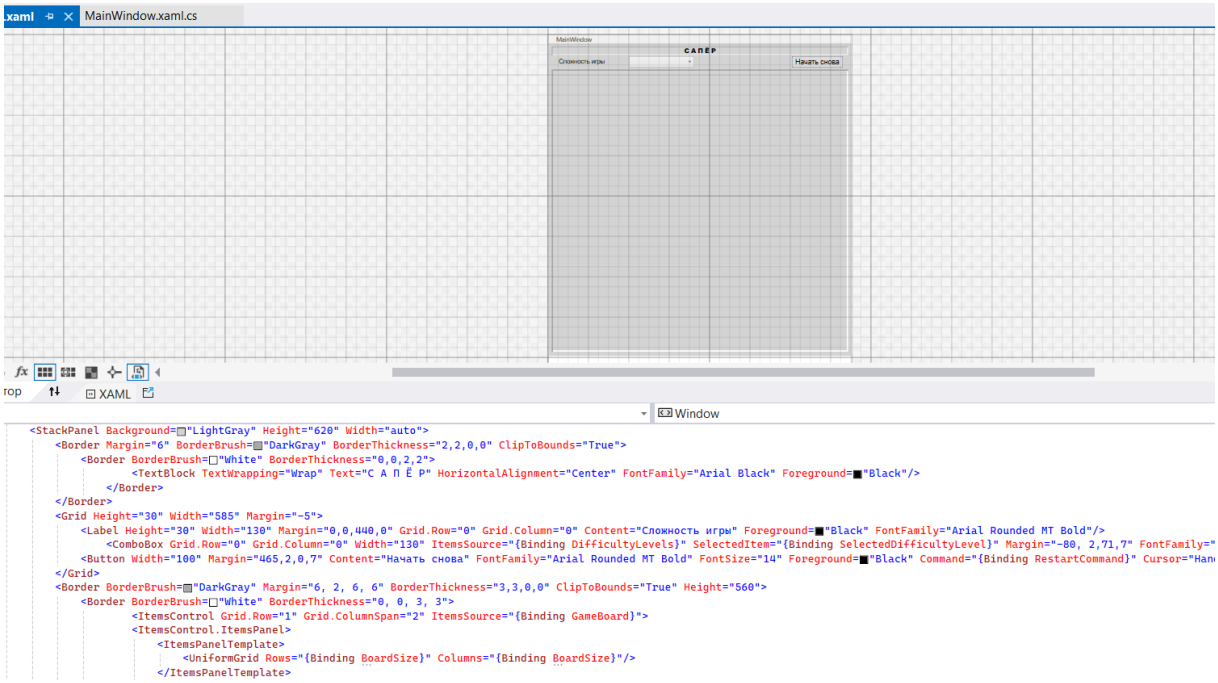


Рисунок 3. Пользовательский интерфейс совместно с текстовой разметкой *XAML*.

В главном окне разметки проекта создано две *StackPanel*, одна содержит обводку пользовательского интерфейса, другая игровое поле и кнопку с связкой чтобы пользователь мог начать заново игру, в котором будет содержаться *Border*. В *Border* находится *ComboBox*, который содержит связку с выбором уровня сложности. Создан новый *Border* который будет содержать связку с элементами игрового поля: размер игрового поля, обработчик нажатия на ячейку и обработчик отображения после нажатия на ячейку (см. рис.3).

# Тестирование

Игра основывается на двух основных действиях: нажатие правой кнопки мыши (ПКМ) и нажатие левой кнопки мыши (ЛКМ). Из возможных действий, это поменять уровень сложности, начать игру заново, открыть ячейку и отметить ячейку флажком.

При запуске игры блокируется расширение окна. Так же предусмотрено, что при запуске игры сразу запускается первый уровень сложности. При многократном нажатии «Начать снова» игра генерирует новые ячейки с указанным уровнем сложности. При изменении уровня сложности всё работает исправно. При открытии ячейки с бомбой, всплывает окно о выборе «Вы проиграли! Начать новую игру?» с кнопками «Да», «Нет». При нажатии на «Да» запускается новая игра. При нажатии на «Нет» игра закрывается. При открытии всех ячеек, которые не содержат бомбу, всплывает окно аналогичным действиям, что при нажатии на бомбу за исключением надписи. Есть единственная проблема при расставлении всех флажков на игровое поле, игра оканчивается победой.

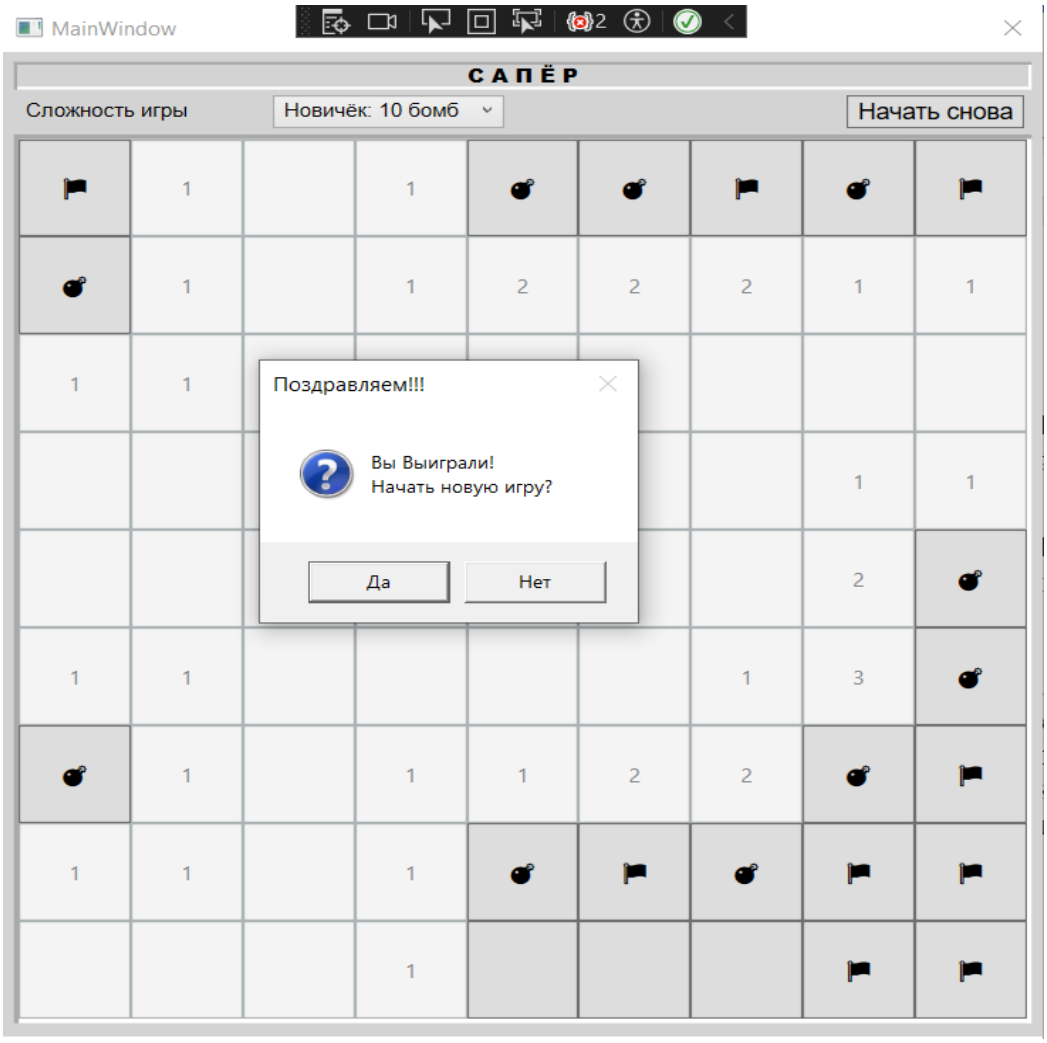


Рисунок 5. Результат игры при тестировании.

# Заключение

В процессе работы создана игра, аналогичная *“MineSweeper”*. Достигнуты желаемые результаты. Изучена работа с текстовой разметкой *XAML*. Соблюдён принцип объектно-ориентированного программирования. Было написано суммарно 500 строк кода.

В целом, данная работа показала эффективность использования объектно-ориентированного подхода при создании компьютерных игр. Результаты исследования могут быть использованы для дальнейшей разработки подобных проектов[5].

# Список используемых источников

1. How To Play Minesweeper // Minesweeper Game URL: https://minesweepergame.com/strategy/how-to-play-minesweeper.php (дата обращения: 20.04.2024).

2. Документация по языку C# // Руководство по C# URL: https://learn.microsoft.com/ru-ru/dotnet/csharp/ (дата обращения: 20.04.2024).

3. Введение в C# // C# и .NET URL: https://metanit.com/sharp/tutorial/ (дата обращения: 20.04.2024).

4. Вайсфельд Мэтт Объектно-ориентированное мышление / Мэтт Вайсфельд. – Санкт-Петербург : Питер, 2014. – 304 с.

5. Нейросеть “GigaChat” генеративная языковая модель версии 4.1 // https://developers.sber.ru/gigachat/