Київський національний університет технологій та дизайну

Лабараторна робота №4

Розробка програмної частини проєкту

Виконав студент БКІ-22

Савенко Ярослав Володимирович

Огризков Роман Олексійович

Крук Павло Миколайович

Романенко Михайло Вадимович

Київ 2024

**Лабораторна робота №4**

**Розробка програмної частини проєкту**

**Мета:** формування практичних навичок організації розробки програмного забезпечення з використанням гнучких (agile) методологій.

**Порядок виконання:**

**1.** Для реалізації проєкту ми обрали мову програмування C# та середовище розробки Unity. Це було стратегічне рішення, оскільки Unity забезпечує зручний інтерфейс для створення ігор, а C# є основною мовою для розробки в Unity.

C# — об'єктно-орієнтована мова програмування, яка дозволяє ефективно працювати з внутрішньою логікою гри та обробляти складні математичні операції, необхідні для функціонування ігрового процесу.

Unity надає широкі можливості для розробки 2D. Використовуючи цю платформу, ми змогли інтегрувати зовнішні ресурси, такі як 2D-моделі, текстури, забезпечуючи візуальне оформлення гри. Зважаючи на те, що ми використовували стандартні бібліотеки Unity, основна увага зосереджувалася на розробці програмного коду гри, що було важливим критерієм для оцінки обсягу виконаних робіт.

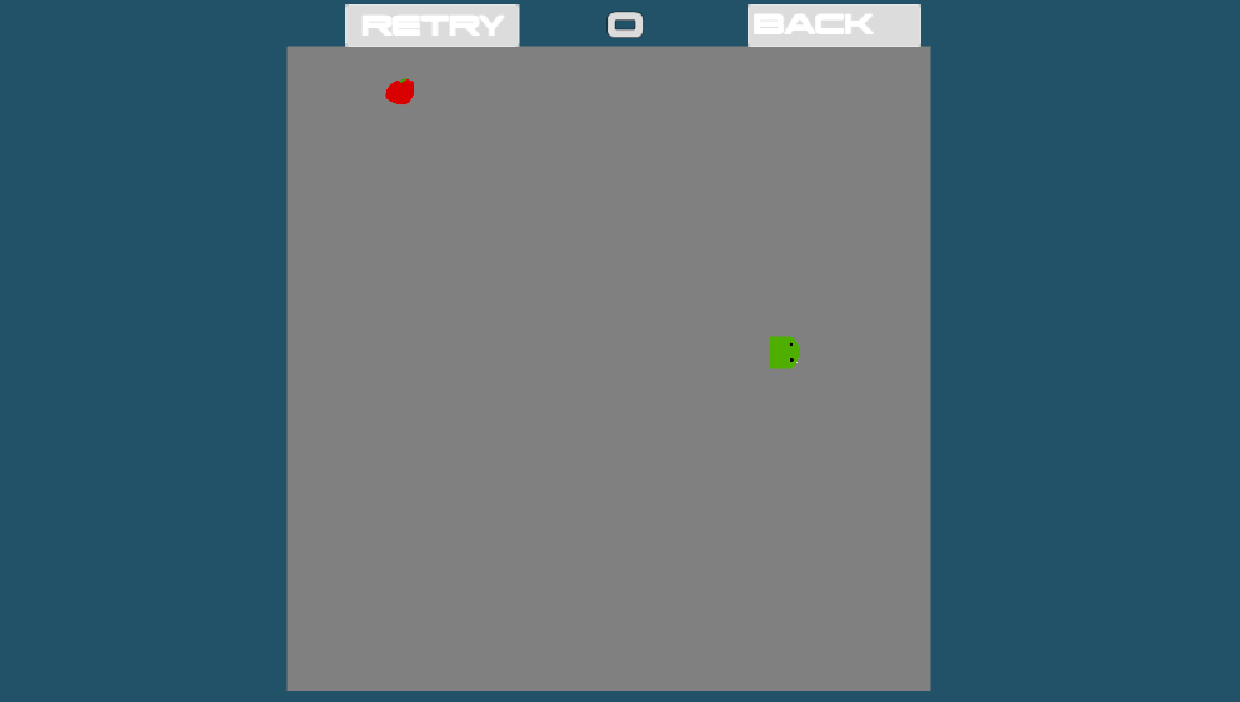
Тому вибір саме цього стеку технологій дав змогу максимально ефективно створити проект із врахуванням вимог і специфіки завдання.

**2.** Відповідно до попередньої лабораторної роботи було реалізовано меню гравця.

Після запуску гри “Змійка” і в нас відкривається вікно гри з головним меню на якому розташовано три кнопки: “Грати”,”Налаштування” ,”Вихід”.



При натисканні кнопки “Грати” відкривається ігрове поле після чого починається розрахунок первиних кординат всіх ігрових об’єктів та визначення розміру ігрового поля та рендеренг ігрового вікна.

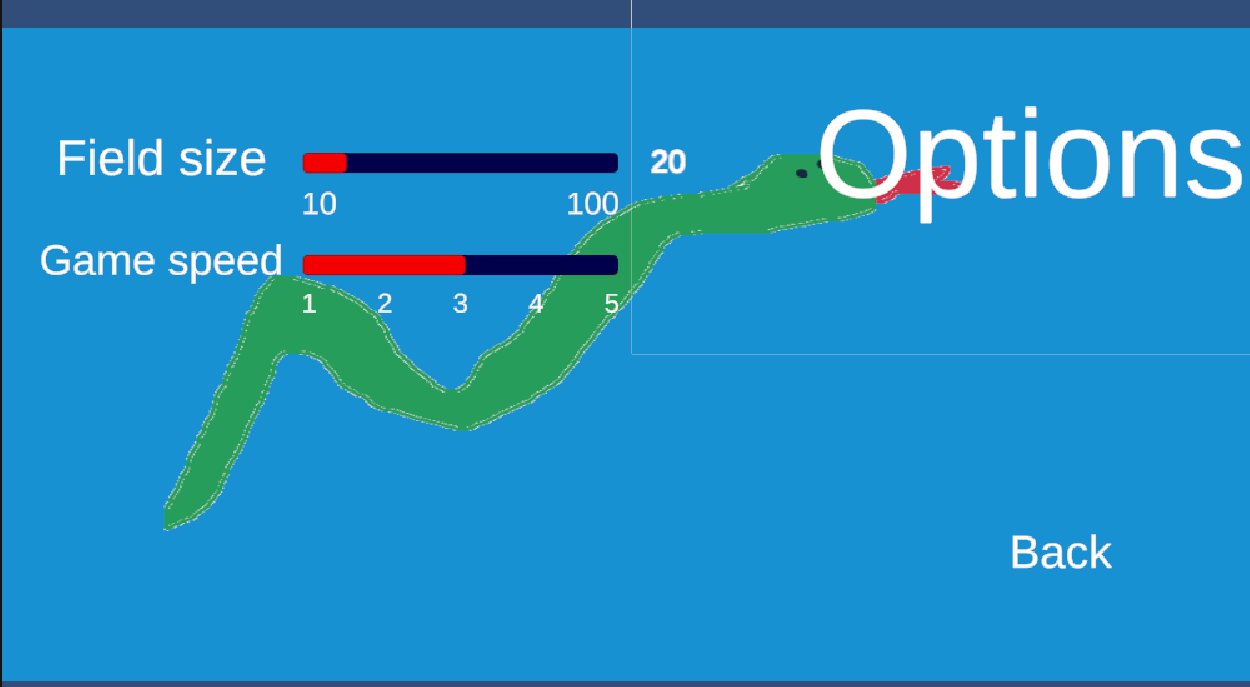


При натискані кнопки “Налаштування” відкривається вікно налаштуваннь, що містить два поля:

1.Розмір поля. Змінюється за допомогою повзунка в межах від 10 до 100.

2.Швидкість руху. Змінюється за допомогою повзунка в межах від 1 до 5.

При натисканні кнопки “Hазад” вікно налаштуваннь закривається і користувач повертається на Головне меню.



Найголовнішими скриптами є Snake, GameHandler, LevelGrid  
Коротко пройдемось по них:

**Snake:**

Цей скрипт реалізує логіку гри «Змійка», контролюючи її рух, обробляючи введення від користувача, перевіряючи зіткнення з їжею та власним тілом. Основні функції включають:

* **Керування рухом**: Змійка рухається по сітці відповідно до напрямку, обраного гравцем.
* **Збільшення розміру**: Коли змійка з'їдає їжу, її розмір збільшується.
* **Перевірка на смерть**: Якщо голова змійки потрапляє в ту ж позицію, що й будь-яка частина тіла, змійка гине.

Скрипт також контролює швидкість руху змійки та візуалізує її частини тіла відповідно до позицій у грі.

**GameHandler :**

Цей скрипт управляє загальною логікою гри, зокрема налаштуванням ігрового поля, веденням рахунку та налаштуванням камери. Основні функції:

**Ініціалізація:** Під час старту гри скрипт налаштовує розмір ігрового поля, з якого береться розмір з налаштувань гри, і ініціалізує сітку (LevelGrid), в якій буде рухатися змійка.

**Керування рахунком:** Ведеться облік очок, які гравець отримує за досягнення (наприклад, за їжу). Є методи для отримання рахунку та його збільшення.

**Налаштування фону та камери:** Скрипт відповідальний за пропорційне масштабування фону та коригування позиції камери, щоб забезпечити правильне відображення ігрового поля.

**Повернення на попередній екран:** Метод для завантаження попередньої сцени, що дозволяє гравцеві повернутися назад у меню.

Таким чином, скрипт «GameHandler» відповідає за організацію ігрового середовища та інтеграцію різних елементів гри.

**LevelGrid**

Цей скрипт управляє логікою ігрового поля, зокрема розміщенням їжі та перевіркою, чи змійка її з'їла. Основні функції:

**Конструктор:** Ініціалізує розміри ігрового поля (width та height).

**Налаштування:** Метод Setup приймає об'єкт змійки і викликає метод для генерації їжі на полі.

**Генерація їжі:** Метод SpawnFood випадковим чином вибирає позицію для їжі на полі, уникаючи зіткнень з позиціями змійки. Створює об'єкт їжі з відповідним спрайтом.

**Перевірка споживання їжі:** Метод TrySnakeEatFood перевіряє, чи змійка з'їла їжу (порівнюючи її позицію з позицією їжі). Якщо так, їжа знищується, генерується нова, і рахунок збільшується.

**Валідація позицій:** Метод ValidateGridPosition перевіряє, чи знаходиться позиція змійки в межах поля, і, якщо це не так, "переносить" її на інший край (роблячи поле безшовним).

Отже, скрипт LevelGrid відповідає за управління ігровим полем, генерацію їжі та обробку взаємодії змійки з їжею.

3. Було розділено розробку проекту на такі єтапи:  
Планування та реалізація діаграм та механік гри (Реалізовував Савенко Ярослав)

Реалізація ігрового процесу (Огризков Роман)

Реалізація меню гри (Романенко Михайло)

Реалізація документацій гри (Крук Павло)

Кожен із етапів розробки не виконувася самостійно всі етапи виконувались під груповим розбором деталей, або з командною роботою певних аспектів.

Кожен із виконавців цього проекту зробив певний фрагмент коду, аби всі учасники команди були залучені саме до кодумання, бо це нашої першочерговою цілью цієї роботи.

4. Використані матеріали:

<https://www.youtube.com/watch?v=Iz22-o7l6bc&list=PLzDRvYVwl53ucaUs0YGfyyKXdgqh5OtiK&ab_channel=CodeMonkey>

**Висновок:** У ході цієї лабораторної роботи ми реалізували гру «Змійка» на основу платформи Unity мовою програмування с#. Кожен етап розробки цього проекту виконування командно, що було основою метою цього проекту