ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи

на ступінь вищої освіт бакалавр

на тему: «**РОЗРОБКА ГРИ MILITARY QUEST ПІД ПЛАТФОРМУ ANDROID У ЖАНРІ "ШУТЕР 2.5D" З ВИКОРИСТАННЯМ ІГРОВОГО РУШІЯ UNITY МОВОЮ C#**»

Виконав: студент 4 курсу, групи ПД–42
спеціальності:
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Олексенко Р.Г.
(прізвище та ініціали)
Керівник Дібрівний О.А.
(прізвище та ініціали)
Рецензент_____
(прізвище та ініціали)

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти - «Бакалавр»
Спеціальність - 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Негоденко О.В.
«____»_____2023 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

ОЛЕКСЕНКО РОМАН ГРИГОРОВИЧ

(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка гри military quest під платформу android у жанрі "шутер 2.5d" з використанням ігрового рушія unity мовою c#»

Керівник роботи доктор філософії Дібрівний О.А.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «24» лютого 2023 року №26.

2. Строк подання студентом роботи «1» червня 2023 року
3. Вихідні данні до роботи:
   3.1. Положення побудови гри;
   3.2. Методи побудови гри;
   3.3. Розробка моделі гри;
   3.4. Науково-технічна література.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно зробити):
   4.1. Загальні положення побудови гри;
   4.2. Аналіз технології і методів побудови гри;

4.3. Висновки

5. Перелік графічного матеріалу:

5.1. Основні характеристики роботи;

5.2. Актуальність задачі;

5.3. Реалізація алгоритмів гри

5.4. Висновки

6. Дата видачі завдання « 25 » лютого 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми бакалаврської роботи	25.02.2023	Виконано
2	Підбір науково-технічної літератури	28.02.2023	Виконано
3	Дослідження аналогів та актуальності додатку	02.03.2023	Виконано
4	Аналіз та вибір інструментів для розробки додатку	05.03.2023	Виконано
5	Проектування та реалізація	29.04.2023	Виконано
6	Вступ, висновки, реферат	12.05.2023	Виконано
7	Розробка обов‘язкових демонстраційних матеріалів	14.05.2023	Виконано
8	Попередній захист	24.05.2023	Виконано
9	Здача роботи	1.06.2023	

Студент_____ Олексенко Р.Г.
(підпис) (прізвище та ініціали)

Керівник роботи_____ Дібрівний О.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи: _____

Ключові слова: Rider, Blender , C#, Unity.

MILITARY QUEST ПІД ПЛАТФОРМУ ANDROID У ЖАНРІ "ШУТЕР 2.5D" З ВИКОРИСТАННЯМ ІГРОВОГО РУШІЯ UNITY МОВОЮ **C#**»

Об’єктом дослідження – ігровий процес у грі в жанрі "shooter".

Предметом дослідження – інструменти створення ігрового додатку в жанрі "shooter" у 2.5D просторі.

Мета роботи – підвищення зацікавленості користувача в грі military quest в жанрі “шутер 2.5d” за рахунок реалізації різноманіття рівнів.

Методи дослідження – Розробка головної ігрової механіки з використанням ігрового рушія Unity та мови програмування C#, включаючи патерни (SOLID, Singleton, Object Pooling, DRY та ін.)

Завдання дослідження, які потрібно виконати:

1. Дослідити ринок ігор для мобільних платформ, визначити популярні жанри та ігрові механіки, що можуть бути успішними серед аудиторії.
2. Розробити концепт гри "Military Quest", визначивши її історію, головну механіку та геймплей.
3. Визначити технічні вимоги до розробки гри, включаючи аспекти графіки, звуку, оптимізації та інші технічні питання.
4. Розробити головну ігрову механіку, включаючи функції управління героєм, бойову систему, рівні складності та інші аспекти геймплею.
5. Провести тестування гри на різних пристроях та платформах, виявити та виправити помилки, доповнити гру корисними функціями.

Галузь використання –

ЗМІСТ

ВСТУП

Актуальність теми. В 21 столітті майже всі грають, або грали в відеоігри, особливо діти. І у зв'язку з коронавірусом гравців стало ще більше. А доказом, що ігри є актуальні – це статистика від Newzoo.

За даними дослідницької компанії Newzoo, світовий ринок відеоігор досяг вартості більше ніж 170 мільярдів доларів у 2020 році. Також варто відзначити, що за останні роки кількість гравців значно зросла. За даними Statista, у 2020 році більше 2,7 мільярдів людей грали відеоігри по всьому світу. І ця кількість зростає щорічно. Таким чином, розробка ігор є досить перспективною галуззю для дослідження та розвитку.

Дохід ігрової індустрії значно зросла в останні роки. За даними Newzoo, в 2021 році глобальний дохід ігрової індустрії складе $189.3 мільярдів, що є на 9.6% більше, ніж у 2020 році. Також, відомо, що ринок мобільних ігор є найбільш прибутковим сегментом з річним доходом в $90.7 мільярдів у 2021 році.

Об'єктом дослідження – ігровий процес у грі в жанрі "shooter".

Предметом дослідження – інструменти створення ігрового додатку в жанрі "shooter" у 2.5D просторі.

Мета роботи – підвищення зацікавленості користувача в грі military quest в жанрі "шутер 2.5d" за рахунок реалізації різноманіття рівнів.

Завдання дослідження, які потрібно виконати:

1. Провести аналіз та обрати жанр відеогри.

2. Розглянути актуальність ігор.

3. Провести аналіз подібних ігрових додатків для усунення недоліків в своєму додатку.

4. Провести аналіз та обрати середовища для розробки гри.

5. Розробити ігровий додаток.

6. Провести тестування відеогри та усунення помилок.

Методика дослідження. Розробка головної ігрової механіки з використанням ігрового рушія Unity та мови програмування C#, включаючи патерни (SOLID, Singleton, Object Pooling, DRY та ін.)

Науковою новизною даного дослідження є поєднання розробки ігрового додатку у жанрі "Shooter 2.5D" з використанням ігрового рушія Unity та мови програмування C#.

Практична значущість результатів дослідження полягає в створенні захоплюючого ігрового додатку для мобільних платформ, який може бути використаний як засіб розваги для користувачів, а також як дослідницький об'єкт для подальших наукових досліджень у галузі розробки ігор та геймдизайну. Результати дослідження можуть бути використані для покращення процесу розробки ігрових додатків та для навчання студентів.

Методика дослідження. Розробка головної ігрової механіки з використанням ігрового рушія Unity та мови програмування C#, включаючи патерни (SOLID, Singleton, Object Pooling, DRY та ін.)

Науковою новизною даного дослідження є поєднання розробки ігрового додатку у жанрі "Shooter 2.5D" з використанням ігрового рушія Unity та мови програмування C#.

Практична значущість результатів дослідження полягає в створенні захоплюючого ігрового додатку для мобільних платформ, який може бути використаний як засіб розваги для користувачів, а також як дослідницький об'єкт для подальших наукових досліджень у галузі розробки ігор та геймдизайну. Результати дослідження можуть бути використані для покращення процесу розробки ігрових додатків та для навчання студентів.

РОЗДІЛ 1 АНАЛІЗ ЖАНРІВ

1.1. Що ж таке відеоігри?

Відеоігра - це електронна гра, яка включає взаємодію користувача з інтерфейсом або пристроєм введення, таким як джойстик, контролер, клавіатура або сенсорний екран. Ця взаємодія дозволяє отримувати візуальну зворотну відповідь на екрані телевізора, монітора комп'ютера, плоского дисплея або сенсорного екрана портативних пристроїв або гарнітури віртуальної реальності.

Відеоігри можуть мати різні форми, включаючи графічні ігри, текстові пригодницькі ігри та комп'ютерні шахи. Більшість сучасних відеоігор мають аудіовізуальний характер і включають звукові ефекти та музику, які відтворюються через динаміки або навушники. Деякі відеоігри також використовують інші типи сенсорного зворотного зв'язку, такі як тактильні відчуття, що дозволяють відчути рухи чи вібрацію пристрою.

Крім того, в деяких відеоіграх можна використовувати мікрофон та веб-камеру для гри, спілкування та прямих трансляцій. Відеоігри можуть бути розроблені для різних платформ, включаючи комп'ютери, консолі, портативні пристрої та віртуальну реальність.

1.2. Класифікація відеоігор

Відеоігри зазвичай класифікуються відповідно до їх апаратної платформи, яка традиційно включає аркадні відеоігри, консольні ігри та комп'ютерні (ПК) ігри; останній також включає в себе LAN-ігри, онлайн-ігри та браузерні ігри. Зовсім недавно індустрія відеоігор розширилася на мобільні ігри за допомогою мобільних пристроїв, систем віртуальної та доповненої реальності та віддалених хмарних ігор.

1.2.1. Аркадні відеоігри

Аркадні відеоігри приймають вхідні дані гравця від своїх органів управління, обробляють їх за допомогою електричних або комп'ютеризованих компонентів і відображають вихідні дані на електронному моніторі або аналогічному дисплеї. Усі аркадні відеоігри оплачуються монетами або приймають інші платіжні засоби, розміщуються в аркадних автоматах та розміщуються в ігрових автоматах поряд з іншими видами аркадних ігор. До кінця 1990-х років аркадні відеоігри були найбільшим і найбільш технологічно продвинутим сегментом індустрії відеоігор.

1.2.2. Консольна гра

Консольна гра - це тип відеоігри, що складається з зображень та часто звуків, що генеруються ігровою консоллю, які відображаються на телевізорі або аналогічній аудіо-відео системі, і якими може маніпулювати гравець. Ця маніпуляція зазвичай відбувається за допомогою портативного пристрою, підключеного до консолі, який називається контролером. Контролер зазвичай містить кілька кнопок та елементів управління напрямком, таких як аналогові джойстики, кожному з яких призначена мета взаємодії з зображеннями на екрані та управління ними. Дисплей, динаміки, консоль та елементи управління консолі також можуть бути об'єднані в один невеликий об'єкт, відомий як портативна гра.

1.2.3. Комп'ютерна гра

Гра для персонального комп'ютера, також відома як комп'ютерна гра або гра для ПК, є електронною грою (зазвичай, відеогрою), яку грають на персональному комп'ютері (ПК). Її визначальні характеристики включають: більш різноманітне та визначуване користувачем геймплейне обладнання та програмне забезпечення; і, як правило, більша ємність введення, обробки, відео та аудіо виведення.

1.2.4. Багатокористувацька відеогра

Багатокористувацька відеогра - це вид відеоігор, у яких можуть брати участь більше одного гравця в одному ігровому середовищі одночасно. Це може стосуватись локальних ігор, де гравці грають на одній консолі або комп'ютері, або мережевих ігор, де гравці можуть грати на різних пристроях через локальну або глобальну мережу, наприклад, через Інтернет.

У багатокористувацьких іграх гравці можуть змагатися один з одним, співпрацювати як команди або спостерігати за діями інших гравців. Це створює можливість для соціальної взаємодії та спілкування між гравцями, що відрізняє їх від одиночних ігор. Багатокористувацькі ігри можуть включати різні жанри, від шутерів до онлайн-рольових ігор, де гравці можуть взаємодіяти у віртуальних світах та спільно вирішувати завдання.

Головна особливість багатокористувацьких ігор полягає в можливості грати разом з реальними людьми, а не з комп'ютером. Це дозволяє створити більш динамічний та соціальний ігровий досвід, де гравці можуть взаємодіяти, спілкуватись та спільно досягати цілей у віртуальному світі гри.

1.2.5. Онлайн-гра

Онлайн-гра - це відеогра, у яку частково або переважно грають через Інтернет або будь-яку іншу доступну комп'ютерну мережу. Онлайн-ігри широко поширені на сучасних ігрових платформах, включаючи ПК, консолі та мобільні пристрої, і охоплюють багато жанрів, включаючи шутери від першої особи, стратегічні ігри та масові багатокористувацькі рольові онлайн-ігри (MMORPG). У 2019 році виручка в сегменті онлайн-ігор досягла 16,9 млрд доларів, з яких 4,2 млрд доларів були отримані в Китаї і 3,5 млрд доларів в США. З 2010-х років загальна тенденція серед онлайн-ігор полягає в тому, щоб використовувати їх як ігри як послугу, використовуючи схеми монетизації, такі як лутбокси та бойові пропуски, як придбані предмети поверх безкоштовно пропонованих ігор. На відміну від придбаних в роздріб ігор, проблема онлайн-ігор полягає в тому, що в них не можна грати постійно, оскільки для їх роботи потрібні спеціальні сервери.

1.2.6. Браузерна гра

Браузерна гра або «флеш-гра» - це відеоігра, в яку можна грати через Інтернет з використанням веб-браузера. Вони в основному безкоштовні і можуть бути для одного гравця або для багатьох гравців.

Деякі браузерні ігри також доступні у вигляді мобільних додатків, ігор для ПК або консолей. Для користувачів перевагою браузерної версії є те, що не потрібно встановлювати гру; браузер автоматично завантажує необхідний контент з сайту гри. Однак браузерна версія може мати менше функцій або гіршу графіку порівняно з іншими, які зазвичай є нативними додатками.

1.2.7. Мобільна гра

Мобільна гра або гра для смартфонів - це відеогра, в яку зазвичай грають на мобільному телефоні. Цей термін також відноситься до всіх ігор, в які можна грати на будь-якому портативному пристрої, включаючи мобільний телефон (мультифункціональний телефон або смартфон), планшет, КПК і портативну ігрову консоль, портативний медіаплеєр або графічний калькулятор, з доступом до мережі або без нього. Найбільш ранньою відомою грою для мобільних телефонів був варіант тетрісу на пристрої Hagenuk MT-2000 1994 року.

1.2.8. Віртуальна реальність (VR)

Віртуальна реальність (VR) - це змодельований досвід, у якому використовується відстеження поз і 3D-дисплеї поблизу очей, щоб надати користувачеві відчуття занурення у віртуальний світ. Додатки віртуальної реальності включають розваги (особливо відеоігри), освіту (наприклад, медичну або військову підготовку) та бізнес (наприклад, віртуальні зустрічі). Інші різні типи технологій у стилі віртуальної реальності включають доповнену реальність та змішану реальність, іноді називають розширеною реальністю або XR, хоча визначення в даний час змінюються через зародження галузі.

1.2.9. Розширена реальність (AR)

Розширена реальність (AR) - це технологія, яка поєднує реальний світ з віртуальним контентом, створюючи інтерактивний досвід. AR дозволяє взаємодіяти з віртуальними об'єктами у реальному часі і в реальному середовищі. Вона використовує різні сенсорні модальності, такі як зорова, слухова, тактильна, соматосенсорна та обонятельна, для створення іммерсивного досвіду.

Основні функції AR включають поєднання реального та віртуального світу, взаємодію в реальному часі та точне тривимірне наложення віртуальних об'єктів на реальні. AR може додавати віртуальний контент до природного середовища, щоб доповнити його, або приховувати реальні об'єкти, створюючи ефект маскування.

AR переплітається з фізичним світом, що дозволяє сприймати віртуальний контент як частину реального середовища. Він змінює спосіб сприйняття навколишнього світу, роблячи його більш іммерсивним. На відміну від віртуальної реальності, яка повністю замінює реальне середовище, AR дозволяє користувачам бачити реальний світ, збагачений віртуальними елементами.

1.3. Жанри відеоігор

Жанр відеоігри - це неформальна класифікація відеоігор, що базується на тому, як в них грають, а не на візуальних або наративних елементах. Це не залежить від сеттінгу, на відміну від художніх творів, які виражені іншими засобами, такими як фільми чи книги. Наприклад, шутер залишається шутером незалежно від того, де і коли він відбувається. Жанр конкретної гри відкритий для суб'єктивної інтерпретації. Окрема гра може належати одразу до декількох жанрів.

Екшн-ігри - це вид відеоігор, які зосереджені на фізичних випробуваннях та вимагають від гравця зорово-моторної координації та рухових навичок для подолання викликів у грі. У цих іграх гравець контролює головного персонажа та здійснює більшість дій.

Екшн-ігри включають різні піджанри. Наприклад, платформери, де гравець керує персонажем, який пересувається по різних платформах та здійснює скакання й стрибки для досягнення мети. Файтинги - це ігри, в яких гравці керують бійцями і здійснюють бойові прийоми та комбінації, змагаючись один з одним.

Один з найпопулярніших піджанрів екшн-ігор - це шутери. Вони вимагають від гравця стрілянини та уміння ефективно керувати зброєю. Шутери стали домінуючим жанром відеоігор з 1990-х років і продовжують зберігати свою популярність.

Екшн-ігри відомі також своїм твіч-геймплеєм, що означає швидкі та реактивні дії, вимагаючи від гравця швидкості та точності у виконанні рухів.

Платформер (часто називається грою "скакуни" або "бігуни") - це піджанр екшн-відеоігор, в яких головною метою є переміщення персонажа гравця між точками в оточуючому середовищі. Платформери характеризуються рівнями, які складаються з нерівної місцевості та підвісних платформ різної висоти, по яких потрібно стрибати та карабкатися, щоб пройти. На геймплей можуть впливати й інші акробатичні маневри, такі як гойдання на ліанах або крюках для захоплення, стрибки зі стін, ривки в повітрі, ковзання по повітрю, вистріли з гармат або стрибки з трампліна чи батута. Ігри, в яких стрибки повністю автоматизовані, наприклад 3D-ігри з серії The Legend of Zelda, не належать до цього жанру.

Шутери - це піджанр відеоігор, в яких головна увага майже повністю зосереджена на поразці ворогів персонажа за допомогою зброї, яку гравець отримує. Зазвичай це вогнепальна зброя або якась інша зброя дальньої дії, і її можна використовувати в поєднанні з іншими інструментами, такими як гранати для непрямого нападу, броня для додаткового захисту або аксесуари, такі як оптичні приціли, для зміни поведінки зброї. Звичайний ресурс, який можна знайти в багатьох шутерах - це боєприпаси, броня або здоров'я, а також поліпшення, які покращують зброю персонажа гравця.

Шутери-герої - це багатокористувацькі шутери від першої або третьої особи, в яких наголошується на заздалегідь розроблених "героїчних" персонажах, кожен з яких має відмінні здібності та / або зброю, характерну для них. Шутери-герої

настійно сприяють командній роботі між гравцями у команді, допомагаючи гравцям вибирати ефективні комбінації персонажів-героїв та координувати використання здібностей героїв під час матчу. Поза матчем у гравців є можливість налаштовувати зовнішній вигляд героїв, але без будь-яких інших ігрових ефектів. Героїчні шутери натхнені жанром багатокористувацьких онлайн-бойових арен та популярними командними шутерами, такими як Team Fortress 2. Приклади шутерів про героїв включають Overwatch, Paladins, Apex Legends та Valorant.

Файтинги - це видеоігри з акцентом на ближньому бою, зазвичай один на один або проти невеликої кількості рівних супротивників, часто з включенням жорстоких і перебільшених беззбройних атак. Більшість файтингів мають велику кількість ігрових персонажів і конкурентний багатокористувацький режим. Хоча більшість файтингів зосереджені на бійці руками, деякі файтинги, такі як Soulcalibur та Samurai Shodown, фокусуються на боях з використанням зброї ближнього бою. Багато бойових ігор містять сильно акцентовані атаки, засновані на різних системах бойових мистецтв. Файтинги були одним із домінуючих жанрів в відеоіграх до кінця 1990-х років, коли цей жанр трохи занепав. Однак цей спад був недовгим, оскільки такі ігри, як Mortal Kombat, домінують у демографії бійців сучасної епохи. Тим не менше, ігри різних піджанрів стають все більш популярними. Super Smash Bros. та її відхилення від традиційного набору правил файтингів - одна з тих ігор, яка здобула велику кількість шанувальників через свій менталітет розробки партійних ігор "веселощі важливіше, ніж форма".

Стелс-гра - це тип відеоігри, в якій гравець в основному використовує хитрість, щоб уникати противників або подолати їх. Ігри цього жанру зазвичай дозволяють гравцю залишатися непоміченим, ховаючись, крадучись або використовуючи маскування. Деякі ігри дозволяють гравцеві вибирати між прихованим підходом або прямою атакою антагоністів, але нагороджують гравця за більш широке використання хитрості. У цьому жанрі використовуються теми шпигунства, боротьби з тероризмом і шахрайством, а головними героями є оперативники спецназу, спеціальні агенти, секретні агенти, злодії, ніндзя або

вбивці. Деякі ігри також поєднують елементи хитрості з іншими жанрами, такими як шутери від першої особи, а також платформери.

У іграх на виживання гравець починає з мінімальними ресурсами в ворожому середовищі відкритого світу і повинен збирати ресурси, створювати інструменти, зброю та укриття, щоб вижити якомога довше. Багато з них відбуваються в процедурно згенерованих середовищах і не мають обмежень за часом. Вони можуть перетинатися з жанром жахів виживання, в якому гравець повинен вижити в надприродному середовищі, такому як зомбі-апокаліпсис.

Ритм-гра, також відома як ритм-екшн, є жанром відеоігор, який поєднує в собі елементи екшну і музики. Основна ідея полягає в тому, щоб викликати у гравця почуття ритму і синхронізації. У цьому жанрі можна знайти як танцювальні ігри, наприклад, Dance Dance Revolution, так і музичні ігри, наприклад, Rock Band і Guitar Hero.

У ритм-грах гравцю потрібно натискати кнопки у відповідний момент часу. На екрані відображаються піктограми або символи, що показують, яку кнопку потрібно натиснути. Гра надає очки гравцю залежно від того, наскільки точно і синхронно він натискає кнопки відповідно до ритму музики. Точність і синхронізація з ритмом є основними факторами, за якими нараховуються очки.

Королівська битва - це популярний жанр відеоігор, який поєднує елементи виживання, дослідження та збору ресурсів з геймплеєм "останній виживший". У цих іграх велика кількість гравців починає гру з мінімальним спорядженням і змушена шукати зброю, броню та інші ресурси для усунення інших учасників і залишання в живих. Головною метою гравця є залишитися в безпечній гральній зоні, яка поступово звужується протягом гри. Виграє останній гравець або команда, що залишається стояти.

Жанр королівської битви отримав значний успіх, зокрема завдяки таким популярним іграм, як PlayerUnknown's Battlegrounds, Fortnite Battle Royale, Garena Free Fire, Apex Legends і Call of Duty: Warzone. Ці ігри привернули мільйони гравців протягом короткого часу після випуску. Вони надають можливість гравцям

випробувати свої навички виживання, тактичного мислення та бойової майстерності в інтенсивних та конкурентних битвах.

Екшн-пригодницькі ігри - це гібридний жанр відеоігор, який поєднує основні елементи жанрів екшн і пригодницьких ігор. Зазвичай, у класичних пригодницьких іграх є ситуаційні задачі, які гравець повинен досліджувати та вирішувати для завершення сюжетної лінії, практично не вимагаючи дій. Якщо ж дія є, то вона зазвичай обмежена окремими випадками. З іншого боку, у класичних екшн-іграх геймплей базується на взаємодіях в реальному часі, які викликають виклик рефлексам гравця та координації очей і рук. Пригодницькі ігри поєднують у собі ці жанри, задіюючи як зорово-моторну координацію, так і навички розв'язання проблем.

Ігри жахів на виживання це жанр, що поєднує страх та напруження з елементами фантастики жахів. Вони створюють налякану атмосферу, залучаючи гравця до світу зомбі, монстрів та обмежених ресурсів. Ці ігри вимагають від гравця вміння виживати, розв'язувати головоломки та збирати ресурси, створюючи незабутній досвід, який доводиться переживати зі страхом і впевненістю власного виживання..

Метроідванія - це піджанр платформерів, який отримав свою назву на честь ігор Metroid і Castlevania. В цих іграх гравцю надається велика світова карта, яку він може досліджувати. Однак, деякі частини світу початково недоступні через закриті двері або перешкоди, які можна пройти лише після отримання спеціальних інструментів, зброї або навичок під час гри.

Отримання цих покращень допомагає гравцеві подолати складніших ворогів, знаходити приховані області та секрети, а також часто вимагає повернень до вже пройдених ділянок карти. Головна особливість метроідваній полягає в їх нелинійному геймплеї, де гравець сам вибирає напрямок дослідження світу і вирішує, коли і як використовувати отримані покращення.

Пригодницькі ігри є жанром відеоігор, в яких гравець виконує роль головного героя в захоплюючій інтерактивній історії. Головна мета цього жанру - дослідження світу гри та вирішення головоломок. Оскільки пригодницькі ігри

фокусуються на наративі, вони широко користуються елементами літератури і кіно, і мають різноманітні жанрові напрямки.

Більшість пригодницьких ігор призначені для одного гравця, оскільки вони ставлять акцент на сюжет та розвиток персонажів, що ускладнює розробку мультиплеєрного режиму гри. Перша відома пригодницька гра, "Колосальна пригода в печері", була випущена в 1976 році, але з того часу з'явилися численні інші відомі серії, такі як "Zork", "King's Quest", "Monkey Island", "Syberia" і "Myst", які продовжують захоплювати уяву гравців своїми захоплюючими історіями та викликами.

Текстові пригоди були одними з перших пригодницьких ігор, відомих також як інтерактивна фантастика. У таких іграх, наприклад у популярній серії Zork, гравець використовував клавіатуру для введення команд, таких як "візьми мотузку" або "йди на захід", а комп'ютер надавав опис того, що відбувається. Велика частина програмування була присвячена аналізу текстових команд від гравця.

Перша графічна пригода, Mystery House для Apple II, використовувала графіку в ранніх домашніх комп'ютерах. З поширенням графіки пригодні ігри стали використовувати візуальні елементи для доповнення або заміни текстових описів (наприклад, зображення поточного місцезнаходження). Початкові графічні пригоди вимагали введення команд за допомогою текстового парсера, але з появою миші з'явився жанр "вкажи та клацни", де гравець може просто клацнути на об'єкти на екрані, наприклад, на мотузку, щоб підняти її.

Візуальна новела - це гра, яка має переважно статичну графіку, зазвичай в аніме стилі. Вона подібна до романів або театральних вистав. Багато візуальних романів мають систему статистики, яку гравець повинен розвивати, щоб продовжувати сюжет, і можуть мати різні кінцівки, що дозволяє більш динамічно реагувати на дії гравця. Часто візуальні новели є симуляторами побачень, включаючи ігри бісьодзьо. Цей жанр особливо популярний в Японії, де він складає майже 70% випущених комп'ютерних ігор. Хоча візуальні новели рідко випускаються для ігрових консолей, деякі популярні ігри іноді портуються на такі

системи, як Dreamcast або PlayStation 2. За межами Японії ринок візуальних новел раніше майже не існував, але успіх Nintendo DS допоміг їм з'явитись і на Заході.

Інтерактивний фільм - це жанр кіно, який з'явився з появою лазерних дисків. Він включає передзняті повнометражні мультфільми або епізоди живих виступів, в яких гравець може керувати деякими діями головного героя. Наприклад, в ситуації небезпеки гравець сам обирає, який хід, дію або комбінацію виконати. У цих іграх гравець обмежений вибором або вгадуванням дій, які розробники підготували. Інтерактивні фільми зазвичай відрізняються від звичайних ігор, де використовується повноекранне відео, FMV. Вони намагаються інтегрувати відео в сам ігровий процес.

Близько у той же період часу з'явилися 3D-пригодницькі ігри в реальному часі. До цих ігор належать Nightfall (1998), Shenmue (1999), realMyst (2000), Shadow of Memories (2001), Uru: Ages Beyond Myst (2003) та франшиза Yakuza (2005). Вони розширили традиційний пригодницький геймплей, додаючи атрибути, які зазвичай пов'язуються з екшн-іграми, наприклад, свободу руху та фізично обґрунтовану поведінку.

Відеоігри-головоломки складають широкий жанр відеоігор, в яких основна увага приділяється вирішенню головоломок. Типи головоломок можуть перевірити навички розв'язання проблем, включаючи логіку, розпізнавання образів, розв'язання послідовностей, просторове розпізнавання та завершення слів. Багато ігор-головоломок включають елемент реального часу та вимагають швидкого мислення, наприклад Tetris (1984) та Lemmings (1991).

Клон прориву (також відомий як "розбиття блоків" або "м'яч і весло") - це підклас жанру головоломок. Цей жанр названий на честь динаміки блоку, керованого гравцем (називається "весло"), на якому базується гра, який ударяє м'ячем по різних об'єктах, таких як кольорові плитки, спеціальні плитки та неруйнівні плитки (називаються "кирпичами"). Термін "руйнівник кирпичів" був вигаданий на початку 2000-х і в основному стосується більш сучасних ігор. Деякими ранніми прикладами є Arkanoid, розроблений Taito в 1986 році, і оригінальний Breakout, розроблений Atari в 1976 році.

Логічна гра Логічні ігри-головоломки демонструють логіку та механізми, які однакові на протязі всієї гри. Їх вирішення зазвичай потребує навичок дедуктивного мислення.

The Splatters, гра для Xbox Live, заснована на фізиці. Фізична гра - це тип логічної відеоігри-головоломки, в якій гравець повинен використовувати гру фізики та оточуюче середовище для вирішення кожної головоломки. Ігри з фізикою використовують послідовну фізику, щоб зробити гру більш складною. Цей жанр особливо популярний в онлайн-флеш-іграх та мобільних іграх. Педагоги використовують ці ігри для демонстрації принципів фізики.

Це логічні головоломки, які вимагають елементів програмування. Прикладами є The Incredible Machine, SpaceChem та Infinifactory.

Цей піджанр включає ігри типу «вказати та клацнути», які часто мають схожість з пригодницькими іграми та симуляторами ходьби. На відміну від логічних головоломок, для вирішення цих ігор зазвичай потрібне індуктивне мислення. Визначальною ознакою є те, що гравець повинен експериментувати з механізмами на кожному рівні, перш ніж зможе їх вирішити. Елементи головоломки часто не мають послідовності протягом усієї гри, тому вимагають відгадування та перевірки, а також дослідження, щоб розкрити більше головоломки. До них відносяться Myst, Limbo, The Dig, Monument Valley та ігри-квести, такі як The Room.

Гра зі схованими об'єктами (іноді називається пригодницькою головоломкою зі схованими об'єктами (HOPA)) - це жанр відеоігор-головоломок, в якому гравець повинен знайти предмети зі списку, приховані в сцені. Ігри зі схованими об'єктами - популярна тенденція в казуальних іграх, і відносно недорога для покупки. Mystery Case Files: Huntsville, випущена Big Fish Games в 2005 році, вважається першою сучасною грою зі схованими об'єктами, що з'явилася на підйомі казуальних ігор в середині 2000-х.

Гра "Покажи зображення" - це тип головоломки, у якому фотографія або зображення поступово розкриваються частинами.

У відеоіграх зі збігом плиток гравець маніпулює плитками, щоб змусити їх зникнути відповідно до критерію зіставлення. Жанр почався з Chain Shot 1985 року! та має схожість з іграми "падаючий блок", такими як тетріс. Цей жанр включає в себе ігри, що потребують заміни фішок, такі як Bejeweled або Candy Crush Saga, ігри, адаптовані до класичної гри Маджонг, що базується на плитках, такі як Mahjong Trails, та ігри, в яких фішки стріляють по дошці, такі як Zuma. У багатьох останніх іграх зі збігом плиток критерієм зіставлення є розміщення заданої кількості плиток одного типу так, щоб вони прилягали один до одного. Це число часто дорівнює трьом, і відповідне підмножина ігор зі збігом плиток називається "іграми зі збігом трьох". Інші приклади включають Threes та Lumines.

Також було випущено безліч цифрових адаптацій традиційних головоломок, включаючи пасьянси та маджонг. Навіть знайомі головоломки зі словами, головоломки з числами та асоціативні головоломки були адаптовані у вигляді ігор, таких як "Тренування мозку доктора Кавасіми".

Головоломки-платформери характеризуються використанням структури платформної гри для керування грою, задачі якої ґрунтуються переважно на головоломках.

Випущена Enix в 1983 році Door Door і випущена Sega в 1985 році Doki Doki Penguin Land (для SG-1000), можливо, є першими прикладами, хоча жанр різноманітний, і класифікації можуть відрізнятися.Doki Doki Penguin Land дозволяла гравцям бігати і стрибати в типовий платформний спосіб, але вони також могли руйнувати блоки, і їм було доручено направити яйце в нижню частину рівня, не давши йому розбитися.

Геймплей рольових відеоігор ґрунтується на основі настільних рольових ігор, зокрема Dungeons & Dragons. У цих іграх гравець приймає на себе роль персонажа, якого можна розвивати, набираючи досвід і збільшуючи його силу протягом гри. Основною метою є подолання складнощів та перемога над монстрами. Гравець отримує очки досвіду за успішне виконання завдань, які відображають прогрес персонажа в його обраній професії або класі і дають змогу отримувати нові навички на певних рівнях. Багато рольових ігор мають великі

відкриті світи, які можна досліджувати. Ці світи зазвичай населені монстрами і включають важливі локації, такі як міста, підземелля та замки. З поширенням домашніх комп'ютерів жанр рольових відеоігор з'явився серед перших і швидко став популярним. Елементи геймплею, такі як розвиток персонажа через отримання досвіду, були використані й у інших жанрах, наприклад, у пригодницьких іграх. Ранні рольові ігри зазвичай працювали у пошаговому режимі, але сучасні рольові ігри все більше розвиваються в реальному часі. Таким чином, жанр рольових ігор відображає тенденцію стратегічних ігор до переходу від пошагового до реального часу у бойових сценах.

Рольова гра в жанрі екшн - це вид рольових відеоігор, що включає бої в реальному часі, а не поштовхові або основані на меню, часто запозичуючи елементи з ігор в жанрі екшн або пригодницьких ігор. Деякі з перших рольових ігор в жанрі екшн були випущені компанією Nihon Falcom в 1980-х роках, наприклад, серія Dragon Slayer та серія Ys. Пізніше так звані «Діаблоклоны» також є частиною цього жанру. Хоча точне визначення жанру змінюється, у типовій рольовій грі наголошується на бою, часто спрощуючи або видаляючи небойові атрибути та статистику, а також їх вплив на розвиток персонажа. Крім того, бій завжди відбувається за допомогою системи реального часу (отже і "дія"), яка ґрунтується на здатності гравця виконувати певні дії зі швидкістю та точністю для визначення успіху, а не в основному використовує атрибути персонажа гравця для визначення цього.

MMORPG (масові багатокористувацькі рольові онлайн-ігри) з'явилися наприкінці 1990-х років як графічний варіант текстових MUD, які існували з 1978 року. Вони не просто дозволяють гравцям розвивати своїх персонажів, але й створюють величезний світ, в якому сотні гравців можуть взаємодіяти один з одним в режимі реального часу. Концепція масових багатокористувацьких ігор швидко розповсюдилася й поєдналася з іншими жанрами. Фантастичні MMORPG, такі як Final Fantasy XI, The Lord of the Rings Online: Shadows of Angmar і The Elder Scrolls Online, є найпопулярнішими представниками цього жанру. Найвідомішою платною грою є World of Warcraft, а серед безкоштовних популярні RuneScape і

TERA. З'являються й інші типи MMORPG. Науково-фантастичні MMORPG, наприклад Phantasy Star Online, становлять меншу частину ринку, з найвідомішим представником - космічною науково-фантастичною грою EVE Online. Існують також інші масові багатокористувацькі онлайн-ігри, які не мають типового RPG сеттінгу, наприклад Second Life і Ingress.

Піджанр відеоігор roguelike запозичує свою назву та елементи геймплею з комп'ютерної гри 1980 року Rogue. На перший погляд, roguelike - це двовимірне крокування по підземеллях з високим рівнем випадковості завдяки процедурній генерації, акцентом на статистичний розвиток персонажа та використання постійної смерті. Хоча традиційно використовується текстовий користувацький інтерфейс, в багатьох таких іграх використовуються графічні плитки для подолання обмежень набору символів. Нові ігри, які відступають від традиційних елементів hack-and-slash, але в іншому зберігають функції процедурної генерації та постійної смерті, іноді називають «шахраями».

РПГ Піджанр тактичних рольових ігор в основному описує ігри, які включають в себе геймплей зі стратегічних ігор як альтернативу традиційним системам РПГ. Ігрок керує кінцевою групою і бореться з такою ж кількістю ворогів, але цей жанр включає в себе стратегічний геймплей, такий як тактичне рухання по ізометричній сітці. Жанр бере свій початок у настільних рольових іграх, де у кожного гравця є час, щоб вирішити дії свого персонажа.

RPG-пісочниця або RPG з відкритим світом надає гравцеві більшу свободу та зазвичай містить кілька більш відкритих світів зі свободним переміщенням (що означає, що гравець не обмежений одним шляхом, обмеженим каменями, парканами і т.д.). Рольові ігри-пісочниці мають схожість з іншими іграми-пісочницями, такими як серія Grand Theft Auto, з великою кількістю інтерактивних неігрових персонажів, великою кількістю контенту та, як правило, одними з найбільших світів для дослідження та найбільш тривалим часом гри серед усіх RPG завдяки вражаючій кількості допоміжного контенту, не критичного для основної сюжетної лінії гри. RPG-пісочниці часто намагаються імітувати весь регіон свого сеттингу. Популярні приклади цього піджанру включають серію

Dragon Slayer від Nihon Falcom, ранні ігри Dragon Quest від Chunsoft, Wasteland від Interplay Entertainment, серії SaGa та Mana від Squaresoft, System Shock 2 від Irrational Games та Looking Glass Studios, Deus Ex від Ion Storm, серії The Elder Scrolls та Fallout від Bethesda Softworks та Interplay Entertainment, Fable від Lionhead Studios, серіал "Готика" від Piranha Bytes та серію Xenoblade Chronicles від Monolith Soft.

DRPG (Dungeon RPG) - це піджанр рольових ігор, в яких гравець керує групою пригодників у пошуках через підземелля або лабіринти, зазвичай представленими у вигляді сітки. У цих іграх головна увага зосереджена на дослідженні підземних просторів. Прикладами таких ігор є відомі серії, такі як Wizardry, Might and Magic, Bard's Tale, Etrian Odyssey і Elminage. Цей піджанр також часто називають "кляксами", оскільки гравець переміщує всю групу персонажів по ігровому полю як одне ціле або "краплю".

Більшість "клякс" ігор є поштовховими, але деякі, наприклад Dungeon Master, Legend of Grimrock і Eye of the Beholder, дозволяють грати в реальному часі. У ранніх іграх цього жанру відсутня автоматична мапа, тому гравцям доводилося самостійно малювати карту, щоб відстежувати свій прогрес у підземеллях. Такі ігри часто містять головоломки, пов'язані з екологією та простором. Наприклад, гравцям може доводитися переміщати камені, щоб відкрити ворота до іншої частини рівня.

Варіант формули рольової гри, в якій гравець набирає монстрів, щоб боротися за них або разом з ними. Зібраних істот часто можна вирощувати або розведення, щоб створювати більш сильних монстрів або підвищувати їх бойові можливості. Прикладом гри з укротителем монстрів є Pokémon.

Відеоігри-симулятори - це різноманітна суперкатегорія ігор, зазвичай призначених для точного моделювання аспектів реальної або уявної реальності. Моделювання будівництва та управління.

Моделювання будівництва та управління (або CMS) - це тип гри-симулятора, в якому гравцям пропонується створювати, розширювати або керувати уявними спільнотами або проектами з обмеженими ресурсами. У міськобудівних іграх

гравець виступає в якості загального планувальника або лідера, щоб задовольнити потреби та бажання ігрових персонажів, ініціюючи структури для їжі, житла, здоров'я, духовної догляду, економічного зростання тощо. Успіх досягається, коли міський бюджет робить зростаючий прибуток, а громадяни отримують покращений спосіб життя щодо житла, здоров'я та товарів. В той час як військовий розвиток часто включається, акцент робиться на економічну потужність.

Ігри-симулятори життя (або ігри зі штучним життям) включають у себе життя або керування одним або кількома штучними життями. Гра-симулятор життя може крутитися навколо людей та взаємин або може бути симуляцією екосистеми. Біологічні симуляції можуть дозволити гравцю експериментувати з генетикою, виживанням або екосистемами, часто у формі освітнього пакета.

Ігри-симулятори транспортних засобів - це жанр відеоігор, які створюють реалістичну інтерпретацію управління різними видами транспортних засобів. Особлива увага зосереджена на моделюванні польоту, де гравець намагається максимально точно керувати літаком. Симулятори бойових польотів є найпопулярнішим піджанром в цій категорії. У таких іграх гравець не лише пілотує літак, але також стикається з бойовими ситуаціями. Крім того, є громадянські авіасимулятори, які фокусуються на цивільних аспектах польоту і не включають елементи бойових дій.

Стратегічні відеоігри зосереджені на геймплеї, який вимагає ретельного та вправного мислення та планування для досягнення перемоги, а дія різноманітна - від світового господарства до тактики на основі загонів. "У більшості стратегічних відеоігор, - каже Ендрю Роллінгс, - гравцю надається божественний погляд на ігровий світ, опосередковано керуючи юнітами, що перебувають під його командуванням". Роллінгс також відзначає, що "походження стратегічних ігор сягає своїми коріннями до їх близьких родичів, настільних ігор".

4X є жанром стратегічних відеоігор, в яких головною метою є eXplore (дослідження), eXpand (розширення), eXploit (експлуатація) та eXterminate (знищення). Ці ігри можуть бути поштовховими або в режимі реального часу. Серія Civilization Сіда Мейера є, можливо, найвідомішим прикладом цього жанру.

Ігри 4X часто охоплюють великий проміжок часу, де гравець має контроль над цивілізацією або імперією. Зазвичай ці ігри мають історичний сеттинг, що охоплює різні епохи людства (наприклад, Empire Earth, Civilization, Golden Age of Civilizations), або науково-фантастичний сеттинг, де гравець керує расою, що панує над галактикою (наприклад, Master of Orion, Galactic Civilizations).

Артилерійські ігри є ранніми комп'ютерними іграми для двох або трьох гравців, в яких танки борються один з одним у бою або подібних ситуаціях. Ці ігри можуть бути в поштовховому режимі та були одними з перших комп'ютерних ігор, що з'явилися. Вони виникли з використання комп'ютерів для військових розрахунків, наприклад, для обчислення траєкторій ракет.

Автобаттлер, також відомий як "автошахи", це жанр стратегічних ігор, що мають подібність до шахів. У цих іграх гравці розташовують персонажів на полі битви у формі сітки на етапі підготовки, а потім ці персонажі автоматично борються з персонажами протилежної команди без безпосередньої участі гравця під час бою. Цей жанр отримав популярність після випуску гри Dota Auto Chess в початку 2019 року, і з того часу було створено багато ігор цього типу від інших розробників, таких як Teamfight Tactics, Dota Underlords та Hearthstone Battlegrounds.

Мультиплеерна онлайн-бойова арена (MOBA) або Action RTS (ARTS) - це жанр стратегічних відеоігор, де кожен гравець керує своїм унікальним персонажем з особливими навичками, які поступово розвиваються протягом гри і сприяють виконанню загальної стратегії команди. Гравці співпрацюють у команді з метою перемоги, яка полягає в знищенні основної структури противника, одночасно захищаючи свою власну базу. Керовані комп'ютером одиниці, які регулярно з'являються і рухаються вздовж певних ліній до бази противника, допомагають гравцям, відомим як "герої" або "чемпіони". Захисні споруди, такі як вежі, завдають шкоди ворожим силам, які намагаються проникнути. Команда перемагає, якщо першою знищує основну споруду противника. Цей жанр поєднує елементи стратегій в реальному часі, рольових ігор та екшн-ігор, проте гравці не будують будівлі або формують одиниці, як у традиційних стратегічних іграх.

Стратегія в реальному часі (RTS) - це жанр комп'ютерних стратегічних ігор, де гравці приймають рішення та здійснюють дії в неперервному режимі, на тлі постійно змінюючогося ігрового стану. Основні характеристики геймплею RTS включають отримання ресурсів, будівництво баз, дослідження технологій і виробництво військових одиниць.

У RTS-іграх гравці змагаються за перемогу, використовуючи стратегічні навички та тактичні рішення. Вони мають збирати ресурси, такі як золото, дерево або камінь, для фінансування своєї економіки і військової сили. Будівництво баз, дослідження нових технологій та тренування військових одиниць дозволяють гравцям розширювати свій вплив на гру світ і готуватися до битви з противниками.

StarCraft, розроблена Blizzard Entertainment, є однією з найпопулярніших RTS-ігор, яка навіть стала елітною дисципліною змагань в Південній Кореї та транслюється по телебаченню. Інші відомі RTS-ігри включають серії Warcraft, Age of Empires, Dawn of War, Company of Heroes, Command and Conquer та Dune II.

У цих іграх гравці повинні бути швидкими в прийнятті рішень, стратегічно мислити та ефективно керувати своїми ресурсами, щоб перемогти противників. Грати в RTS-ігри можна як проти комп'ютера, так і в режимі мультиплеєра, змагаючись з іншими гравцями з усього світу.

Тактика в реальному часі (скорочено RTT, і рідше називається "стратегією реального часу з фіксованими юнітами") - це піджанр тактичних варгеймів, в які грають в реальному часі, що імітують розмірковування та обставини оперативної війни та військових дій. Він також іноді вважається піджанром стратегії в реальному часі, і тому в цьому контексті він може існувати як елемент геймплею або як основа всієї гри. Він відрізняється від геймплею стратегії в реальному часі відсутністю мікрокерування ресурсами, створення бази або підрозділу, а також більшим значенням окремих підрозділів та акцентом на складній тактиці полі бою. Приклади назв включають Warhammer: Dark Omen, World In Conflict, серію Close Combat та ранні тактичні рольові ігри, такі як Bokosuka Wars, Silver Ghost та First Queen.

Ігри Tower Defense мають просту структуру геймплею. У цих іграх контрольовані комп'ютером монстри рухаються по заданому маршруту, а гравець повинен розміщувати або будувати вежі вздовж цього шляху, щоб убивати ворогів. Деякі вежі розташовуються на шляху ворогів, тоді як інші можуть перешкоджати руху ворогів або змінювати їхній маршрут. У більшості ігор Tower Defense різні вежі мають унікальні можливості, такі як отруєння ворогів або їх уповільнення. Гравцю нараховують гроші за вбивство ворогів, які можна використовувати для придбання більшої кількості веж або для покупки покращень, які підвищують потужність або дальність дії вежі.

Термін "покрокова стратегія" (TBS) використовується для визначення певного типу комп'ютерних стратегічних ігор, які відрізняються від стратегій в реальному часі. У покрокових стратегіях гравцю надається час на аналіз ситуації перед здійсненням ходу або дії. В деяких іграх також може бути встановлена обмежена кількість ходів або дій на кожен хід. Хоча цей жанр включає безліч стратегічних ігор, які не є виключно покрокові, ігри цього типу можуть також мати інші функції, які не пов'язані з покроковими аспектом. Деякими прикладами покрокових стратегій є Civilization, Heroes of Might and Magic, Making History і Master of Orion.

Військові ігри - це піджанр стратегічних відеоігор, в яких основна увага приділяється стратегічній або тактичній війні на карті. Військові ігри зазвичай приймають одну з чотирьох архетипних форм, залежно від того, чи є гра поштовховою чи в реальному часі та чи сфокусована гра на військовій стратегії чи тактиці.

Велика стратегічна військова гра - це військова гра, в якій основна увага приділяється великій стратегії: військовій стратегії на рівні руху та використання ресурсів всього національного держави або імперії. Прикладом цього є франшиза Hearts of Iron.

Існують різні типи гоночних ігор, в яких гравець змагається з часом або противниками, використовуючи ті чи інші засоби пересування. Піджанри включають гоночні симулятори (Gran Turismo) та картингові гонки (Mario Kart).

Інші серії гоночних симуляторів, такі як Forza, є одними з найпопулярніших ігор у цій категорії, але класичні аркади, такі як Pole Position, також сюди включаються.

Спортивні ігри імітують заняття традиційними фізичними видами спорту. Деякі з них акцентують увагу на виконанні самого спорту, тоді як інші - на стратегії, що лежить в основі цього виду спорту (наприклад, Championship Manager). Деякі ігри висміюють спорт, щоб створити комічний ефект (наприклад, Arch Rivals). Одна з найбільш продаваних серій в цьому жанрі - серія FIFA. Цей жанр з'явився на початку історії відеоігор (наприклад, Pong) і залишається популярним до цього часу. Деякі інші ігри, такі як NBA Jam, висміюють цей жанр, тоді як інші, наприклад, серії Madden NFL та NBA 2K, намагаються відтворити реалізм та передати відчуття живого спорту.

Спортивні файтинги - це ігри, які повністю відповідають визначенням жанру файтингів і спортивних ігор, таких як відеоігри про бокс та боротьбу. Як такі, їх зазвичай розміщують в окремі піджанри. Часто бої набагато реалістичніші, ніж в традиційних файтингах (хоча ступінь реалізму може сильно відрізнятися), і в багатьох з них представлені франшизи або бійці з реального світу. Приклади цього включають серії Fight Night, UFC і WWE 2K.

1.4.Актуальність ігор

Відеоігри настільки широко поширені серед американських підлітків, що намалювати портрет типового підлітка-геймера — значить відобразити в дзеркалі всіх підлітків в цілому. Практично кожен підліток хоча б якось грає в ігри, незалежно від статі, віку чи соціально-економічного статусу. Можливостей для гри достатньо: підлітки самі володіють кількома гральними пристроями і грають у гри на пристроях, належних друзям, до яких у них немає особистого доступу. Не тільки всезагальний доступ до ігор, але й часті ігри становлять частину типового повсякденного досвіду для половини всіх підлітків. І якщо підліток грає в ігри в певний день, він або вона, мабуть, проведе в них майже годину. Розуміння природи грального процесу життєво важливо для розуміння того, як майже кожен

американський підліток проводить принаймні частину свого дня. Майже всі підлітки грають у ігри. Половина підлітків, які грають у ігри, роблять це щодня.

Майже всі американські підлітки віком від 12 до 17 років, а саме 97%, грають у комп'ютерні, веб-, консольні або мобільні ігри. Підлітки також грають у ці ігри з відносною частотою та тривалістю. Майже третина (31%) геймерів-підлітків грають у ігри щодня, а кожний п'ятий (21%) грає у ігри від трьох до п'яти днів на тиждень.

Половина підлітків, які грають у ігри, роблять це щодня. Крім опитування підлітків про те, як часто вони грають в ігри протягом тижня, Pew Internet Gaming and Civics Survey запитував гравців-підлітків, грали вони в ігри "вчора" і, якщо так, то на який час. У будь-який день 50% геймерів-підлітків повідомляють, що грають у ігри. Приблизно половина тих, хто грав у гру "вчора" (або 24% усіх геймерів-підлітків), кажуть, що грали протягом години. Ще 13% геймерів-підлітків кажуть, що вони грали дві години, а 13% - три години або більше. Раса, етнічна належність та сімейний дохід не впливають на тривалість гри підлітків.

Майже всі дівчата та хлопці грають в відеоігри. Хлопці повідомляють, що грають в ігри частіше та довше, ніж дівчата. У 2008 році стереотип про те, що відеоігри грають лише хлопці, далекий від правди, оскільки дівчата становлять великий (якщо не рівний) процент загальної кількості геймерів: 99% хлопців і 94% дівчат грають в ігри. Хоча майже всі дівчата, так само як і майже всі хлопці, грають в відеоігри, хлопці зазвичай грають у гри частіше та триваліше, ніж дівчата. Хлопці значно частіше грають в ігри щодня, ніж дівчата: 39% хлопців повідомляють про щоденну гру, тоді як 22% дівчат повідомляють про це. Хлопці також частіше, ніж дівчата, грають в ігри в будь-який конкретний день (60% хлопців порівняно з 39% дівчат), і коли хлопці грають, вони грають триваліші періоди часу. Серед підлітків, які грали в ігри «вчора», хлопці та дівчата з однаковою ймовірністю повідомляють, що вони грали протягом години або менше. Однак хлопці вдвічі частіше повідомляють, що грають по дві години і більше щодня, при цьому 34% хлопців грають по дві години і більше на день; 18% дівчат грають в ігри стільки часу.

Молодші підлітки - найбільш запальні геймери. Вік підлітка є також важливою змінною для передбачення частоти та тривалості гри, а також типів ігор та досвіду під час гри. На відміну від більшості інших онлайн- або цифрових видів діяльності, де підлітки старшого віку частіше беруть участь у діяльності, підлітки старшого віку рідше, ніж молодші підлітки, грають в ігри в звичайний день. Вчора більше половини (54%) дітей віком від 12 до 14 років грали в ігри, а серед дітей віком від 15 до 17 років - 46%.

Користувачі широкосмугового доступу дещо частіше грають протягом більш тривалих періодів часу, ніж підлітки, які проживають в будинках без широкосмугового доступу.

Тоді як користувачі комутованого та широкосмугового доступу з однаковою ймовірністю будуть грати в ігри близько години, користувачі широкосмугового доступу з більшою ймовірністю повідомлять, що грають протягом двох годин; 14% повідомляють, що грають так довго в звичайний день, порівняно з 8% користувачів, які грали протягом двох годин "вчора". У загальному, 28% користувачів широкосмугового доступу грають в ігри дві години та більше в звичайний день, тоді як 20% користувачів, які користуються комутованим доступом, роблять те саме.

Щоденний геймер: молодий, чоловік і грає в онлайн-ігри. У контексті публічних дискусій про потенційні позитивні та негативні наслідки гри в ігри, у цьому звіті поставлено завдання розкрити інформацію про підлітків, які часто грають в ігри і, таким чином, частіше піддаються впливу відеоігор. Хто ці заядлі гравці?

Більше третини (31%) підлітків говорять, що грають в ігри щодня. Підлітки, які грають щодня, належать переважно до хлопців: 65% чоловіків та 35% жінок. Ці гравці також молодші: 57% тих, хто грає в ігри щодня, - люди віком 12-14 років, а інші 43% щоденних гравців - у віці 15-17 років. Щоденні геймери з більшою ймовірністю грають на портативних ігрових пристроях (73% щоденних гравців використовують портативні ігрові пристрої порівняно з 57% тих, хто грає рідше) і

так само, як і інші геймери, грають на комп'ютерах, консолях або на мобільному телефоні.

Щоденні гравці частіше повідомляють, що вони грають в онлайн-ігри та грають в ігри з іншими людьми через інтернет. Щоденні гравці частіше, ніж інші гравці (20% проти 12%), заявляють, що вони "найчастіше" грають в ігри з іншими людьми, з якими вони пов'язані через Інтернет. Аналогічним чином, щоденні гравці також частіше повідомляють, що грають в ігри в складі гільдії або групи (так роблять 50% щоденних гравців порівняно з 38% менш частих гравців). Щоденні гравці також трохи частіше грають в ігри в одиночку (87% проти 79%), ніж ті, хто грає рідше, але з такою ж імовірністю, як і будь-який інший геймер, грають в ігри з людьми в одній кімнаті. Щоденні гравці частіше грають в ігри як з людьми, яких вони вперше зустріли особисто, так і з людьми, яких вони вперше зустріли в Інтернеті, ніж з іншими гравцями.

Часті гравці не відчувають себе відокремленими від суспільства. Традиційно геймер часто асоціюється з одиноким ігровим ніком, який віддає перевагу проводити свій час, граючи в відеоігри, а не проводячи його з друзями. Наш опитування спростувало цей стереотип, показавши, що найбільш запалені та активні гравці так само комунікабельні та соціально включені, як і менш активні гравці. Єдиним винятком є текстові повідомлення: щоденні гравці трохи рідше, ніж ті, хто грає рідше, щоденно надсилають або отримують текстові повідомлення як засіб спілкування з друзями. Гравці надсилають текстові повідомлення, але рідше роблять це щодня, ніж ті, хто грає рідше; 32% гравців, які щодня надсилають текстові повідомлення своїм друзям, щодня надсилають текстові повідомлення своїм друзям, тоді як 41% гравців, які грають рідше, щодня надсилають текстові повідомлення друзям.

Проте, крім обміну текстовими повідомленнями, щоденні гравці, так само як і підлітки, які грають рідше, використовують інші способи спілкування з друзями (стаціонарні телефони, мобільні телефони, миттєві повідомлення, електронна пошта та обмін повідомленнями в соціальних мережах) та проводять час з друзями особисто - обличчям до обличчя.

1.5. Вибір жанру

Гравці люблять грати в шутери (FPS) та ігри з видом зверху (top-down) з різних причин, і ось декілька з них:

Адреналін: Шутери і ігри з видом зверху часто мають високий темп та динамічність, що може викликати адреналінову реакцію у гравця. Це дозволяє відчути себе більш захоплюючою інтерактивність гри та приносить задоволення від успішних моментів.

Виклик: Шутери та ігри з видом зверху можуть бути викликом для гравців, особливо на високих рівнях складності. Це вимагає від гравця стратегічного мислення та вміння приймати рішення в умовах обмеженого часу, що може бути стимулом для гравця до поліпшення власних навичок.

Геймплей: Шутери та ігри з видом зверху можуть мати різні геймплейні механіки, що дозволяє гравцеві знаходити свій стиль гри. Шутери можуть бути більш напруженими та екшн-орієнтованими, засновані на швидкій реакції та точній стрільбі. З іншого боку, ігри з видом зверху можуть бути спокійнішими та більш стратегічними, з більшою увагою на планування та управління ресурсами.

Онлайн-гра: Шутери та ігри з видом зверху часто мають онлайн-режим, що дозволяє гравцеві змагатися з іншими гравцями по всьому світу. Це може бути стимулом для гравця до поліпшення власних навичок та досягнення нових досягнень.

Розважальна інтерактивність: Шутери та ігри з видом зверху можуть бути більш розважальними та інтерактивним ими, оскільки вони дозволяють гравцеві відчути себе частиною гри. Гравці можуть зануритися у віртуальний світ та відчути себе як справжні воїни чи керівники. Ігри з видом зверху можуть дозволяти гравцеві керувати великими арміями або бізнесами, що може бути цікавим інтерактивним досвідом.

Інтерактивність з друзями: Шутери та ігри з видом зверху можуть бути приємними для гри з друзями. Онлайн-режими дозволяють гравцеві змагатися та

співпрацювати з друзями, що може створювати незабутні спогади та зміцнювати дружбу.

Великий вибір: Шутери та ігри з видом зверху мають великий вибір гральних платформ та жанрів, що дозволяє гравцеві знайти саме те, що йому подобається. Вони можуть бути настільними, мобільними або на консолях, а також мати різні теми та сюжетні лінії.

Збільшення навичок: Гра в шутери та ігри з видом зверху може допомогти гравцю розвивати різні навички, такі як швидкість реакції, стратегічне мислення та прийняття рішень в умовах обмеженого часу. Ці навички можуть бути корисними у реальному житті.

Розвиток креативності: Деякі шутери та ігри з видом зверху можуть дозволяти гравцеві створювати свої власні рівні та сценарії гри. Це може стимулювати креативність гравця та дозволяти йому виразити свої ідеї через гру.

Навчальні аспекти: Шутери та ігри з видом зверху можуть мати навчальні аспекти , що можуть бути корисними для дітей та дорослих. Наприклад, ігри про історію або науку можуть допомогти вивчати нові теми та розширювати знання. Деякі ігри можуть навчати вмінню співпрацювати з іншими гравцями та розвивати комунікативні навички. Крім того, деякі ігри можуть допомогти у навчанні мов, особливо якщо гра має різноманітні мовні версії.

Розвага та відпочинок: Останнім аспектом, але не менш важливим, є те, що гра в шутери та ігри з видом зверху може бути просто хорошою розвагою та засобом відпочинку. Граючи улюблену гру, гравець може забути про повсякденні проблеми та відпочити.

Загалом, ігри в жанрі шутерів та з видом зверху мають багато позитивних аспектів, які можуть бути корисними для розвитку різних навичок та дозвілля. Проте, як і з будь-якою іншою діяльністю, важливо не перевищувати міру та зберігати баланс між грою та іншими аспектами життя.

1.6. Аналіз подібних ігрових додатків

"Dead Nation" - це гра в жанрі шутер з видом зверху, де гравець повинен боротися з тисячами зомбі. Гравець може покращувати своє озброєння та техніку, які він використовує, щоб відбити напад ворожих сил та захистити людство від зомбі-апокаліпсису.

У грі є можливість грати як один, так і з друзями в кооперативному режимі. Гра має багато рівнів складності, які дозволяють гравцям насолоджуватися грою на різному рівні виклику.

Гра має красиву 2D-графіку та завзяту музику, які доповнюють атмосферу гри та роблять її ще більш захоплюючою.

Zombie Anarchy - ця гра з командними битвами між гравцями та зомбі, які прагнуть знищити людство. Гравець повинен побудувати свою базу та розвивати її, щоб зах истити її від нападів зомбі та інших гравців.

У грі є можливість створення команди з іншими гравцями, які допоможуть у боротьбі зі злом. Кожен гравець має свої унікальні можливості та навички, які допоможуть команді перемогти.

Гра має красиву 3D-графіку та звуковий супровід, який допомагає створити атмосферу постапокаліптичного світу. Гравці можуть боротися зі зомбі в різних місцях, таких як вулиці міста, склади та інші локації.

Killing Floor: Calamity - це шутер від першої особи, де гравець бореться з хвилями зомбі в науковому дослідницькому центрі. Гравці можуть вибирати з різних персонажів з унікальними навичками та зброєю.

Гра має багато рівнів складності, які дозволяють гравцям насолоджуватися грою на різному рівні виклику. Гравці можуть покращувати своє озброєння та здібності, щоб стати сильнішими та вижити в цьому постапокаліптичному світі.

Графіка:

"Dead Nation":

Плюси: Деталізована та красива графіка.

Мінуси: Може викликати проблеми з швидкодією на менш потужних комп'ютерах або мобільних пристроях.

"Zombie Anarchy":

Плюси: Деталізована та красива графіка.

Мінуси: Може викликати проблеми з швидкодією на менш потужних комп'ютерах або мобільних пристроях.

"Killing Floor: Calamity":

Плюси: Графіка високої якості з деталізованими деталями та гарним освітленням.

Мінуси: Може викликати проблеми з швидкодією на менш потужних комп'ютерах або мобільних пристроях.

Геймплей:

"Dead Nation":

Плюси: Захоплюючий та нелегкий геймплей зі спробою вижити в апокаліптичному світі.

Мінуси: Може бути монотонним після декількох годин гри.

"Zombie Anarchy":

Плюси: Великий світ, який можна досліджувати та вдосконалювати, з можливістю збирати ресурси та взаємодіяти з інш гравцями.

Мінуси: Гра може бути складною для новачків, а також може мати проблеми з балансом між гравцями, що може здатися несправедливим.

"Killing Floor: Calamity":

Плюси: Захоплюючий та нелегкий геймплей з багатою історією та хорошим мультиплеєром.

Мінуси: Гра може бути дуже вимогливою для гравців зі слабкими комп'ютерами, а також може викликати стрес через високу складність та кількість ворожих сил.

Зброя та екіпірування:

"Dead Nation":

Плюси: Широкий вибір зброї та екіпірування, яке можна вдосконалювати.

Мінуси: Деякі види зброї можуть бути непрактичними або мало ефективними в деяких ситуаціях.

"Zombie Anarchy":

Плюси: Широкий вибір зброї та екіпірування, яке можна вдосконалювати.

Мінуси: Деякі види зброї та екіпірування можуть бути непрактичними або мало ефективними в деяких ситуаціях.

"Killing Floor: Calamity":

Плюси: Широкий вибір зброї та екіпірування, яке можна вдосконалювати.

Мінуси: Деякі види зброї можуть бути менш ефективними у боротьбі з деякими типами ворожих сил.

Мультиплеєр:

"Dead Nation":

Плюси: Є можливість грати з друзями в кооперативному режимі або викликати їх на битву в режимі PvP.

Мінуси: В режимі PvP можуть виникати проблеми з балансом між гравцями, що може здатися несправедливим.

"Zombie Anarchy":

Плюси: Можливість грати з друзями в кооперативному режимі або викликати їх на битву в режимі PvP.

Мінуси: В режимі PvP можуть виникати проблеми з балансом між гравцями, що може здатися несправедливим.

"Killing Floor: Calamity":

Плюси: Можливість грати з друзями в кооперативному режимі або змагатися з ними в режимі PvP.

Мінуси: Можуть виникати проблеми з підключенням до мультиплеєрної гри, що може зменшити задоволення від гри.

Ціна:

"Dead Nation":

Плюси: Недорога ціна.

Мінуси: Вища ціна порівняно з іншими іграми у цьому списку.

"Zombie Anarchy":

Плюси: Безкоштовна гра з можливістю внутрішніх покупок.

Мінуси: Покупки можуть бути обмежені захоплюючим геймплеєм, можливість витрачати багато грошей на внутрішні покупки, що може створювати дисбаланс між гравцями.

"Killing Floor: Calamity":

Плюси: Розумна ціна порівняно з іншими іграми у цьому списку.

Мінуси: Не безкоштовна гра.

Висновки до розділу 1

Після аналізу жанру шутерів відеоігор та огляду аналогів "Dead Nation", "Zombie Anarchy" і "Killing Floor: Calamity", можна зробити наступні висновки:

"Dead Nation": Цей шутер з перспективою зверху, в якому гравцеві доведеться боротися з хвилею зомбі. Плюси включають атмосферний світ, насичену геймплейну механіку та кооперативний режим для гри з друзями. Однак, можливий мінус - обмежені можливості у виборі різноманітності зброї та навичок персонажа.

"Zombie Anarchy": Ця гра також пропонує гравцеві битися з зомбі, але з перспективою з третьої особи. Переваги цієї гри включають різноманітність зомбі-ворогів, побудову оборонних споруд та можливість використовувати спеціальні навички. Однак, можливий недолік - деякі гравці можуть вважати геймплейну механіку монотонною на довгострокову перспективу.

"Killing Floor: Calamity": Ця гра є спін-оффом відомої серії "Killing Floor" і пропонує гравцеві аркадні зомбі-стрілянини. Перевагами гри є швидкий та інтенсивний геймплей, різноманітність ворогів та можливість використовувати різні типи зброї. Однак, можливий недолік - відсутність глибокого сюжету та обмежені можливості взаємодії з навколишнім середовищем.

Загалом, жанр шутерів пропонує широкий вибір ігор з різними аспектами та насолодою від битв з ворогами. Кінцевий вибір залежить від особистих вподобань гравця, таких які стилі гри, наявність мультиплеєра та особливості геймплею. Наприклад, "Dead Nation" пропонує атмосферний та напружений досвід боротьби

з хвилею зомбі, а "Zombie Anarchy" дозволяє будувати оборонні споруди та використовувати спеціальні навички.

Крім того, слід враховувати, що суб'єктивні вподобання кожного гравця можуть впливати на сприйняття плюсів та мінусів. Що для одного гравця може бути плюсом, для іншого може стати недоліком. Отже, важливо зробити вибір відповідно до своїх особистих пристрастей та очікувань.

В цілому, вибір між "Dead Nation", "Zombie Anarchy" та "Killing Floor: Calamity" залежить від ваших уподобань щодо геймплею, графіки, механіки та інших факторів. Рекомендується ознайомитися з оглядами та відгуками на ці ігри, а також спробувати демо-версії або переглянути геймплейні відео, щоб зробити більш обґрунтований вибір.

РОЗДІЛ 2 ЗАСОБИ РОЗРОБКИ

2.1. Розбір ігрових рушіїв

Ігровий движок - це програмне забезпечення для створення та розробки відеоігор. Ігрові движки зазвичай мають кілька компонентів, які обробляють різні аспекти процесу розробки, такі як графіка, звук, матеріали, мережева взаємодія та фізика.

Найкращий ігровий движок для початківців надає розробникам доступ до тих самих інструментів, якими користуються професійні розробники ігор, і дозволяє створювати складні ігри набагато швидше та простіше, ніж без них. Для тих, хто шукає найкращі ігрові движки, є безліч чудових варіантів.

Важливість найкращих ігрових движків

Найкращий ігровий движок для новачків буде залежати від типу гри, яку вони збираються розробляти, оскільки деякі движки краще підходять для певних жанрів, ніж інші.

При виборі найкращих ігрових движків слід враховувати такі фактори, як якість графіки, масштабованість, бюджет та складність. У 2023 році буде доступно безліч найкращих ігрових движків, які забезпечать відмінну продуктивність та гнучкість для компанії-розробника ігор.

2.2. Найкращий ігровий движок для початківців у 2023 році

Unity

Unity - провідна у світі платформа для створення та роботи з інтерактивним 3D-контентом в реальному часі. Від розробників ігор до художників, архітекторів, дизайнерів автомобілів, режисерів та багатьох інших - всі використовують Unity, щоб втілити свої фантазії в життя. Платформа Unity надає повний набір програмних рішень для створення та роботи з 2D- та 3D-контентом в реальному

часі для різних платформ, включаючи мобільні телефони, планшети, ПК, консолі, а також пристрої доповненої та віртуальної реальності.

Unreal Engine

Компанія Epic Games, заснована в 1991 році, є створювачем серій ігор Unreal, Gears of War та Infinity Blade. Сьогодні Epic розробляє Paragon, Fortnite, SPYJiNX та новий Unreal Tournament. Технологія Unreal Engine від Epic використовується командами будь-якого розміру для створення візуально захоплюючих, високоякісних ігор та додатків для ПК, консолей, VR та мобільних платформ. Розробники також обирають Unreal Engine для візуалізації, дизайну, лінійних розваг та моделювання. Підпишіться на @UnrealEngine в Твіттері та безкоштовно завантажте Unreal Engine 4 на сайті unrealengine.com.

2.3. Найкраще середовище розробки мовою C#

Visual Studio

Візуальна студія (Visual Studio) - це інтегроване середовище розробки (IDE), що створене компанією Microsoft для розробки програмного забезпечення. Ця платформа дозволяє програмістам створювати різноманітні програми для платформ Windows, Android, iOS, Xbox та інших. У Візуальній студії є багато різноманітних інструментів, що допомагають розробнику зосередитись на написанні коду, замість витрачання часу на налаштування середовища. Візуальна студія також має можливість підключення до різних сервісів, таких як GitHub, щоб спростити процес роботи з кодом.

Один з головних переваг Візуальної студії полягає в тому, що вона надає широкі можливості для розробки програмного забезпечення на різних мовах програмування. В середовищі можна розробляти на мовах програмування C#, C++, Visual Basic .NET, Python та інших. Крім того, середовище має багато інструментів для тестування та налагодження коду, що дозволяє програмістам зробити код більш якісним та надійним.

Ще одна важлива перевага Візуальної студії полягає в її можливостях для роботи з відкритим кодом. Вона має вбудовані інструменти для роботи з Git, що дозволяє розробникам легко працювати з репозиторіями та зберігати свій код в хмарі. Більше того, Візуальна студія підтримує платформу .NET, що дозволяє розробникам створювати програмне забезпечення для різних платформ, включаючи Windows, macOS, Linux та інші.

Rider

Rider - це інтегроване середовище розробки (IDE), що розроблене компанією JetBrains для розробки програмного забезпечення на платформі .NET. Rider дозволяє розробникам створювати програми на мовах програмування, таких як C#, VB.NET, F# та інших, включаючи технології ASP.NET, .NET Core і Xamarin. За допомогою Rider розробники можуть створювати як настільні, так і веб-програми, мобільні додатки та інші програмні продукти.

Однією з головних переваг Rider є підтримка розширення ReSharper, що дозволяє розробникам використовувати широкий набір інструментів для покращення якості та продуктивності свого коду. Rider також має вбудовану систему контролю версій, що дозволяє розробникам працювати з репозиторіями, такими як Git, Subversion та Mercurial.

Ще однією важливою перевагою Rider є його підтримка платформи .NET Core, що дозволяє розробникам створювати програмне забезпечення для різних платформ, включаючи Windows, macOS, Linux та інші. Rider також має вбудовану підтримку Docker, що дозволяє розробникам легко створювати, розгортати та керувати контейнеризованими додатками. Крім того, Rider надає можливість розробникам використовувати різні інструменти для тестування та налагодження коду, що дозволяє зробити код більш якісним та надійним.

2.4. Вибір ігрового рушія

Unity - один із найпопулярніших ігрових движків у світі, і це не випадково. Unity має кілька переваг перед іншими ігровими двигунами.

1.Кросплатформність. Unity підтримує понад 25 платформ, включаючи iOS, Android, Windows, Mac, Linux, PlayStation, Xbox та багато інших. Це означає, що ви можете розробити гру для декількох платформ одночасно, що значно спрощує процес розробки та розширює аудиторію.

2.Простота використання. Unity має інтуїтивно зрозумілий інтерфейс та просту систему скриптування мовою C#. Це робить його доступним навіть для новачків у галузі розробки ігор.

3.Відкритий вихідний код. Unity надає відкритий доступ до вихідного коду, що дозволяє розробникам налаштовувати та розширювати двигун, щоб відповідати їхнім потребам.

4.Багатий магазин активів. Unity Asset Store містить понад 30 000 безкоштовних та платних активів, включаючи моделі, текстури, звуки, анімації та багато іншого. Це дозволяє розробникам економити час та ресурси, використовуючи готові ресурси, замість створення їх самостійно.

5.Потужні інструменти створення ігрового контенту. Unity має інструменти для створення ігрового контенту, такі як системи частинок, фізичного моделювання, системи світла та тіні, а також інструменти для створення ігрових інтерфейсів та штучного інтелекту.

6.Активна спільнота. Існує величезна спільнота розробників Unity, які готові допомогти новачкам та ділитися своїми знаннями. Це робить Unity доступнішим і загальнодоступнішим для всіх, хто хоче створювати ігри.

2.5. Вибір середовища розробки

Rider - це інтегроване середовище розробки для платформи .NET, яке надає розробникам можливість створювати програми на мовах програмування, таких як

C#, VB.NET, F# та інших. Він має безліч переваг, які роблять його одним з найкращих виборів для розробників, які працюють з платформою .NET.

Однією з головних переваг Rider є його швидкодія. Це дозволяє розробникам швидко створювати, налагоджувати та розгортати програмне забезпечення на платформі .NET. Крім того, Rider надає розробникам можливість працювати з різними типами проектів, включаючи проекти ASP.NET, .NET Core та Xamarin.

Ще однією перевагою Rider є його інтуїтивно зрозумілий інтерфейс користувача. Це дозволяє новим користувачам легко навчатися використовувати інструмент та швидко починати роботу. Крім того, Rider має багато корисних функцій, таких як автодоповнення коду, рефакторинг та багато інших.

Rider також має вбудовану підтримку розширення ReSharper. Це дозволяє розробникам використовувати широкий набір інструментів для покращення якості та продуктивності свого коду. Наприклад, ReSharper надає можливість автоматичного вирішення проблем з кодом, додавання вихідних коментарів та перегляду статистики використання коду.

Іншою перевагою Rider є підтримка платформи .NET Core, що дозволяє розробникам створювати програмне забезпечення для різних платформ, включаючи Windows, macOS, Linux та інші. Розробники можуть легко створювати, розгортати та керувати контейнеризованими додатками за допомогою Rider.

І нарешті, ще однією з переваг Rider є його відкритість та розширюваність. Rider має відкритий вихідний код, що дозволяє розробникам вносити власні зміни та вдосконалювати середовище розробки. Крім того, Rider має велику спільноту розробників, які надають підтримку та створюють корисні розширення для платформи.

Узагалі, Rider є потужним та зручним інструментом для розробки програмного забезпечення на платформі .NET. Його швидкодія, інтуїтивно зрозумілий інтерфейс, розширюваність та підтримка різних типів проектів роблять його одним з кращих виборів для розробників, які працюють з платформою .NET. Розробники можуть використовувати Rider для створення високоякісного та ефективного програмного забезпечення для різних платформ.

Висновки до розділу 2

Були описані аспекти, які стосуються засобів розробки в індустрії відеоігор. Оцінювання ігрових рушіїв та вибір програмного забезпечення є важливими кроками для успішної розробки відеоігор.

Найкращий ігровий движок для початківців у 2023 році: Unity та Unreal Engine є двома популярними ігровими рушіями, які заслуговують на увагу початківцям. Unity є часто рекомендованим вибором для новачків завдяки своїй легкості вивчення та широкій підтримці. Unreal Engine, з іншого боку, надає потужні інструменти для створення графічно вражаючих ігор, але може бути трохи складнішим для початківців. Вибір між цими двома залежить від вашого досвіду, потреб і перспектив вашого проекту.

Найкраще середовище розробки мовою C#: Visual Studio та Rider є популярними інтегрованими середовищами розробки (IDE) для мови C#. Visual Studio, розроблений компанією Microsoft, є широко використовуваним інструментом з багатьма функціями та підтримкою розробки великих проектів. Rider, випущений компанією JetBrains, також є потужним IDE з підтримкою C# та інших мов програмування. Вибір між ними може залежати від вашої особистої вподоби та попереднього досвіду з використанням цих інструментів.

Вибір ігрового рушія: При виборі ігрового рушія, такого як Unity чи Unreal Engine, слід враховувати такі фактори, як ваша цільова платформа (наприклад, ПК, консолі, мобільні пристрої), потреби вашого проекту, наявність необхідних функцій та ресурсів, які пропонують ці рушії, а також ваш рівень досвіду з використанням цих інструментів. Урахування цих факторів допоможе зробити оптимальний вибір для вашої конкретної ситуації.

РОЗДІЛ 3. РОЗРОБКА ГРИ

3.1 Структура файлів гри

В папкі "assets" знаходяться різні файли гри, такі як зображення, звуки, текстури та інші ресурси. Є папка меню, де знаходиться сцена меню і скріпти для неї. Також є папка "levels" для зберігання ігрових рівнів. Папка зі скріптами взаємодії, деякі скріпти я зберігаю в інших папках для зручності. Наприклад в папці зомбі зберігаються тільки скрипти які виключно для зомбі.

Ще там є файли з асетс стору. В яких є папки "textures" де зберігаються текстури, які використовуються в грі, а в папкі "sounds" - звукові ефекти та музика. Так само там зберігаються різні об’єкти і ефекти.В грі є всього 26 скріптів, які надають логіку грі.

3.2Опис логіки гри

3.2.1. Логіка меню

Є можливісь обрати зброю і вибір рівня. Спочатку всі рівні й зброя закриті, окрім початкового. Відкривається далі після проходження рівнів.

3.2.2. Логіка рівня

Щоб пройти рівень, потрібно виконати певні умови для перемоги.

Умови для перемоги на якомусь з рівнів:

1) Знищити n кількість зомбі.

2) Пройти в певну зону на рівні.

3) Протриматись n кількість часу.

4) Вбити боса рівня.

Поразка настає коли здоров’я гравця падає до 0.Після проходження рівня відкривається наступний рівень.

3.2.3. Логіка стрільби

Коли гравець націлюється на ворога, то йде стрільба. Є певна зона над ворогом, коли гравець розпочне стрільбу. Якщо скінчаться набої, то потрібно перезарядити зброю. Після чого, гравець може далі вести стрільбу.

3.2.4. Логіка монстрів

Звичайні монстри та Бос патрулюють зону й переміщуються по заданій траекторії. Коли в зону монстра зайде гравець, то монстер почне бігти за гравцем та атакувати коли підбіжить.

Зомбі в рівні, де потрібно протриматись певний час, вони йдуть відразу на гравця.

3.2.1. Логіка атаки монстра

Коли він доганяє гравця, є маленька зона й коли гравеець в ній, монстр атакує.Вмикається об’єкт на руці, й коли він попадає по гравцю то гагоситься n кількість урона.

3.3 Опис класів гравця

3.3.1. SystemHP

Клас SystemHP успадковується від класу MonoBehaviour. Він призначений для управління здоров'ям об'єкта в ігровому середовищі.

У класі є три внутрішні змінні: hp (тип int), maxHp (тип int) і _animator (тип Animator), які можуть бути змінені в інспекторі Unity, оскільки вони позначені атрибутом [SerializeField]. Також є статична подія OnZombieDestroyed (тип Action), яка виникає, коли зомбі знищується.

У методі Start() ініціалізується поточне здоров'я об'єкта значенням maxHp. Метод TakeHit(int damage) зменшує поточне здоров'я об'єкта на задане значення damage і, якщо здоров'я стає меншим або рівним нулю, виконується анімація смерті (якщо об'єкт - зомбі), або завантаження нульової сцени (якщо об'єкт - гравець).

Методи LoadSceneWithDelay(float delay) і DestroyAfterAnimation(float delay) представляють собою корутини, які затримують виконання на певну кількість часу, вказану в параметрі delay, а потім виконують дії всередині себе: у першому випадку - завантажують нульову сцену, у другому - знищують об'єкт і викликають подію OnZombieDestroyed. Приклад зображено (рис. 3.1.) та (рис. 3.2.)

Рис 3.1. Смерть зомбі, коли його хп падає до 0

Рис 3.2. Смерть гравця, коли його хп падає до 0\

3.3.2. PlayerController

Клас PlayerController відповідає за керування гравцем в грі. У ньому оголошені змінні для доступу до Rigidbody гравця, джойстика для керування, швидкості руху і швидкості повороту. У методі FixedUpdate() виконується зчитування вхідних даних з джойстика, обчислення вектора напрямку руху та його нормалізація. Якщо вектор напрямку має значення більше 0.1f, то гравець рухається в напрямку джойстика за допомогою функції MovePosition() і повертається в напрямку руху за допомогою функції Lerp(). Клас дозволяє гравцю поворачуватися в напрямку руху.

3.3.3. PlayerMover

Клас PlayerMover відповідає за рух головного героя в грі за допомогою мобільного джойстика. У ньому оголошені змінні для доступу до аніматора, джойстика для керування, швидкості руху і Rigidbody. У методі Start() виконується ініціалізація змінних. У методі FixedUpdate() викликається метод GetMobileInput(), який отримує вхідні дані з джойстика. В методі GetMobileInput() обчислюється вектор напрямку руху за допомогою значень горизонтальної та вертикальної осей джойстика. Якщо вектор напрямку має значення більше 0.5f, то гравець рухається в напрямку джойстика за допомогою функції AddForce(). Крім того, змінюються значення параметрів аніматора Speedy і Speedx, що відповідають за швидкість руху героя по вертикалі та горизонталі відповідно. Клас дозволяє гравцеві рухатися в усіх напрямках з різною швидкістю і допомагає створити реалістичну анімацію руху. Приклад зображено (рис. 3.3.)

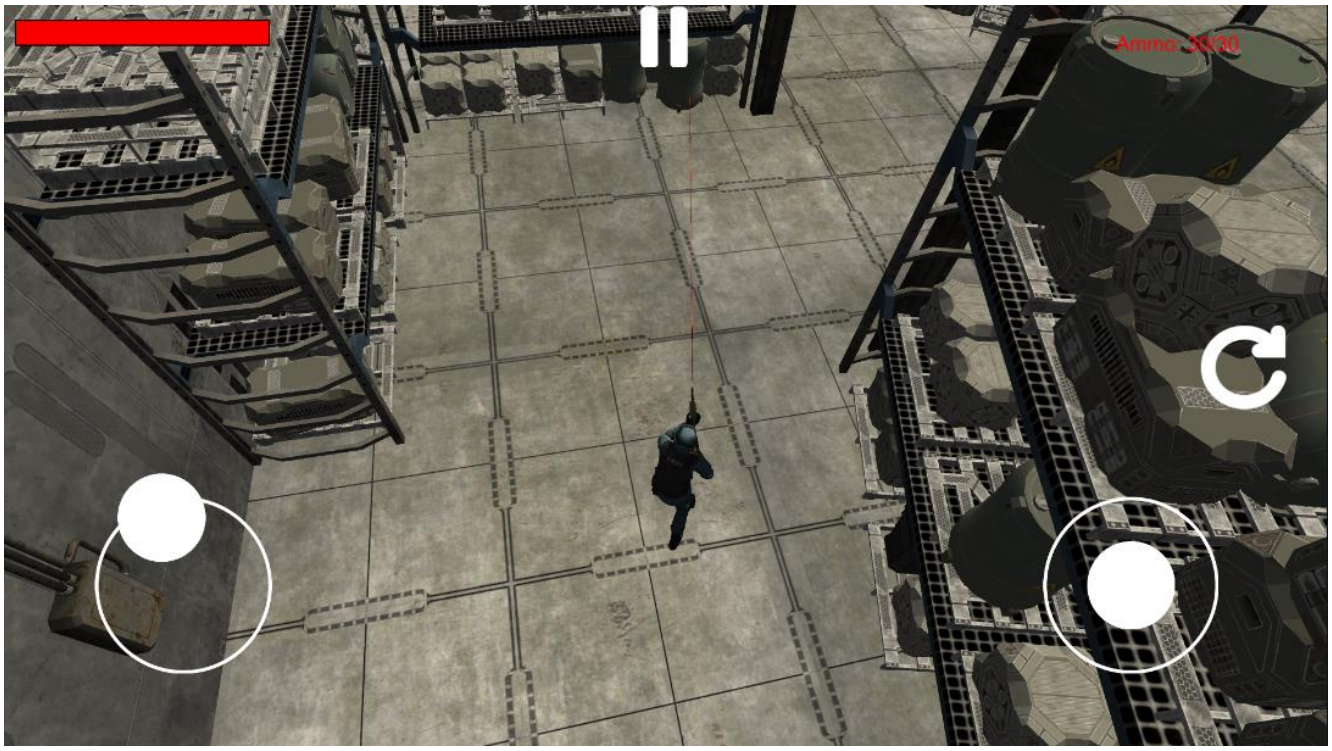


Рис 3.3. Рух гравця

3.3.4. Weapon

Клас Weapon, є скриптом для компоненту GameObject у грі, який представляє зброю, яку може використовувати гравець. У класі є оголошені змінні для налаштування зброї, такі як позиція вистрілу, тег об'єкту з яким зіткнулись, затримка вистрілу, максимальна кількість боєприпасів, аніматор, частинка ефекту вистрілу, аудіо джерело та аудіо кліп. Також є приватні змінні, такі як поточна кількість боєприпасів та час, що пройшов з останнього пострілу. У методі Start() виконується ініціалізація змінних, а саме налаштування звукового джерела та поточної кількості боєприпасів. У методі Update() відбувається перевірка часу, що пройшов з останнього пострілу, та зупинка ефекту вистрілу, якщо час перевищує затримку вистрілу. У класі також є метод OnTriggerStay(), який викликається, коли інший об'єкт зіштовхується з об'єктом з цим скриптом. У методі проводиться перевірка тегу іншого об'єкта та часу, що пройшов з останнього пострілу та наявності боєприпасів. Якщо умови виконуються, викликається метод Shoot(), який відповідає за постріл, звук та візуальний ефект вистрілу. Також метод Shoot() відповідає за видачу об'єкту кулі за допомогою ObjectPool, який повертає вільний

об'єкт з пулу об'єктів. У класі також є метод UpdateAmmoText(), який відповідає за оновлення тексту з кількістю боєприпасів на екрані. Клас міститьдекілька методів для налаштування параметрів зброї, такі як SetAmmoCount() для встановлення кількості боєприпасів, SetFireDelay() для встановлення затримки між вистрілами та SetMaxAmmo() для встановлення максимальної кількості боєприпасів.

Також у класі є метод Reload(), який відповідає за перезарядку зброї. У методі проводиться перевірка наявності боєприпасів та перевірка, чи зброя не вже перезаряджається. Якщо умови виконуються, то відтворюється звук перезарядки, встановлюється флаг перезарядки та запускається таймер на час перезарядки. Після закінчення таймера, кількість боєприпасів встановлюється на максимальне значення.

Окрім того, клас містить метод StopShooting(), який зупиняє вогонь зброї. Цей метод викликається при відпусканні кнопки вогню.

Нарешті, у класі є метод StartShooting(), який запускає вогонь зброї. Цей метод викликається при натисканні кнопки вогню. В методі проводиться перевірка, чи зброя не перезаряджається, чи не вичерпано кількість боєприпасів та чи не перевищено затримку між вистрілами. Якщо умови виконуються, то запускається таймер для затримки між вистрілами та відтворюється звук вистрілу.

Усі ці методи дозволяють налаштувати та керувати зброєю в грі та забезпечують реалістичний ефект стрільби та перезарядки зброї.

3.3.5. Bullet

Клас Bullet відповідає за рух кулі в грі та її повернення до пулу об'єктів.
У класі оголошені змінні для встановлення швидкості руху кулі та доступ до компонента Rigidbody, який відповідає за фізичну поведінку кулі. У методі FixedUpdate() встановлюється вектор швидкості руху кулі у напрямку передньої частини об'єкта (transform.forward) зі швидкістю, встановленою у змінній speed.

Клас Bullet також реалізує інтерфейс IPoolable, що дозволяє повернути кулю до пулу об'єктів для подальшого використання. Для цього в класі реалізовані

методи removeBullet() та Reset(). Метод removeBullet() викликає метод Reset() через 2 секунди, що встановлено за допомогою функції Invoke(). Метод Reset() викликає подію OnReturnToPool, що повідомляє інший об'єкт про повернення кулі до пулу.

Цей клас може бути використаний для реалізації механіки вогневої зброї в грі, де об'єкти можуть випускати кулі, які летять у певному напрямку з певною швидкістю, а після певного часу повертаються до пулу для подальшого використання.

3.3.6. SystemDamage

Клас SystemDamage відповідає за нанесення шкоди іншим об'єктам у грі при зіткненні з ними. У класі оголошені змінні для встановлення значення шкоди, яку буде нанесено, та тег, до якого об'єкта може бути застосована шкода.

У методі OnCollisionEnter() виконується перевірка тегу об'єкта, з яким відбулося зіткнення. Якщо тег співпадає з встановленим тегом _collisionTag, то з об'єкта, з яким відбулося зіткнення, отримується компонент SystemHP, і на нього застосовується функція TakeHit() з встановленою кількістю шкоди _collisionDamage.

Цей клас може бути використаний для реалізації механіки бойової системи в грі, де об'єкти можуть наносити одне одному шкоду при зіткненні.

3.4 Опис класів меню

3.4.1. PauseMenu

Клас PauseMenu відповідає за паузу гри. У ньому оголошені змінні для доступу до панелі паузи та кнопки, яка викликає паузу. У методі Pouse() здійснюється зупинка гри, відображення панелі паузи та приховання кнопки. Метод Exit() викликається при натисканні кнопки "Вихід" на панелі паузи та прибирає панель паузи та відображає кнопку паузи. Метод continuegame()

відповідає за продовження гри, встановлює часовий масштаб на 1, сховання панелі паузи та відображення кнопки паузи. Приклад зображено (рис. 3.4.)

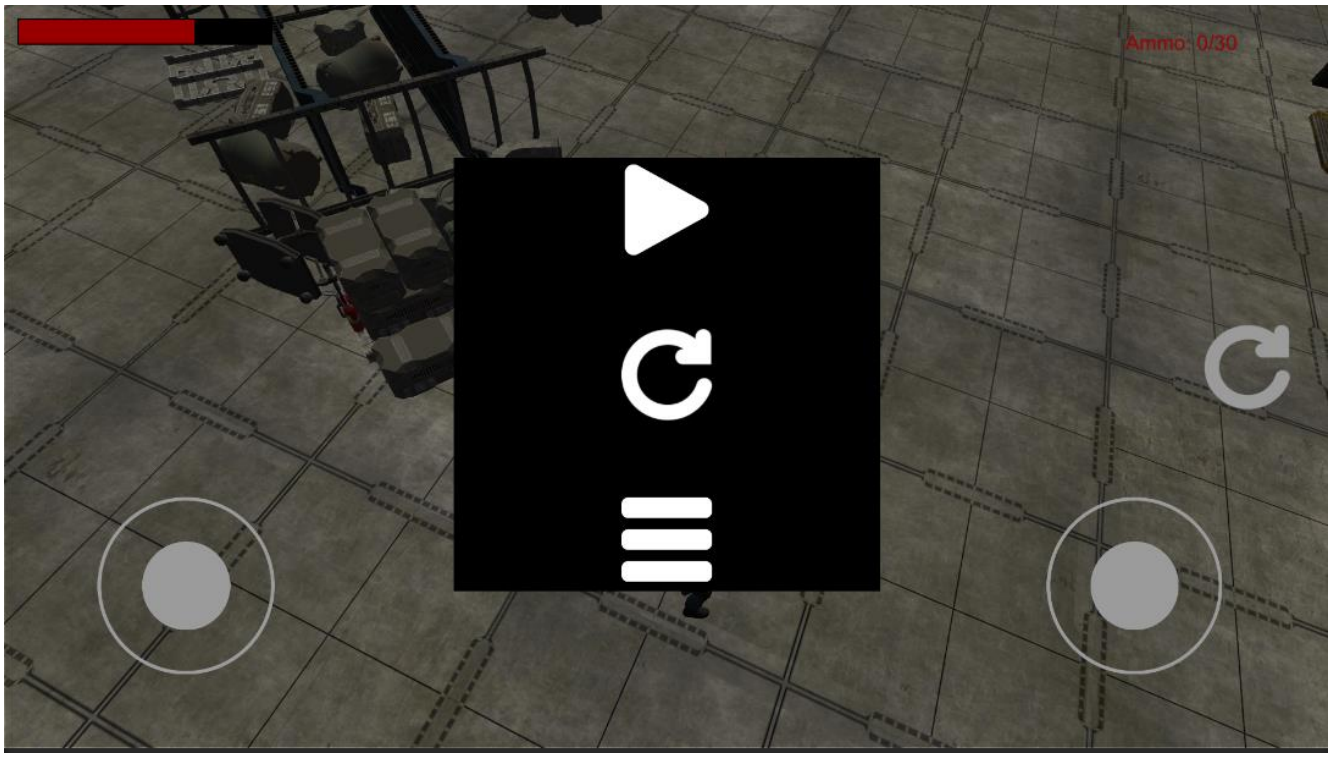


Рис 3.4. Меню паузи

3.4.2. ButtonMenu

Клас ButtonMenu містить метод gotoMenu(), який викликається при натисканні кнопки, і відповідає за перехід до головного меню гри. У методі використовується SceneManager для завантаження сцени з індексом 0 (яка зазвичай відповідає за головне меню), а також Time.timeScale встановлюється на 1, щоб позбутися ефекту зупинки гри, якщо він був активований в попередній сцені. Змінна button використовується, ймовірно, для збереження посилання на саму кнопку, але не використовується в цьому класі. Приклад зображено (рис. 3.5.)

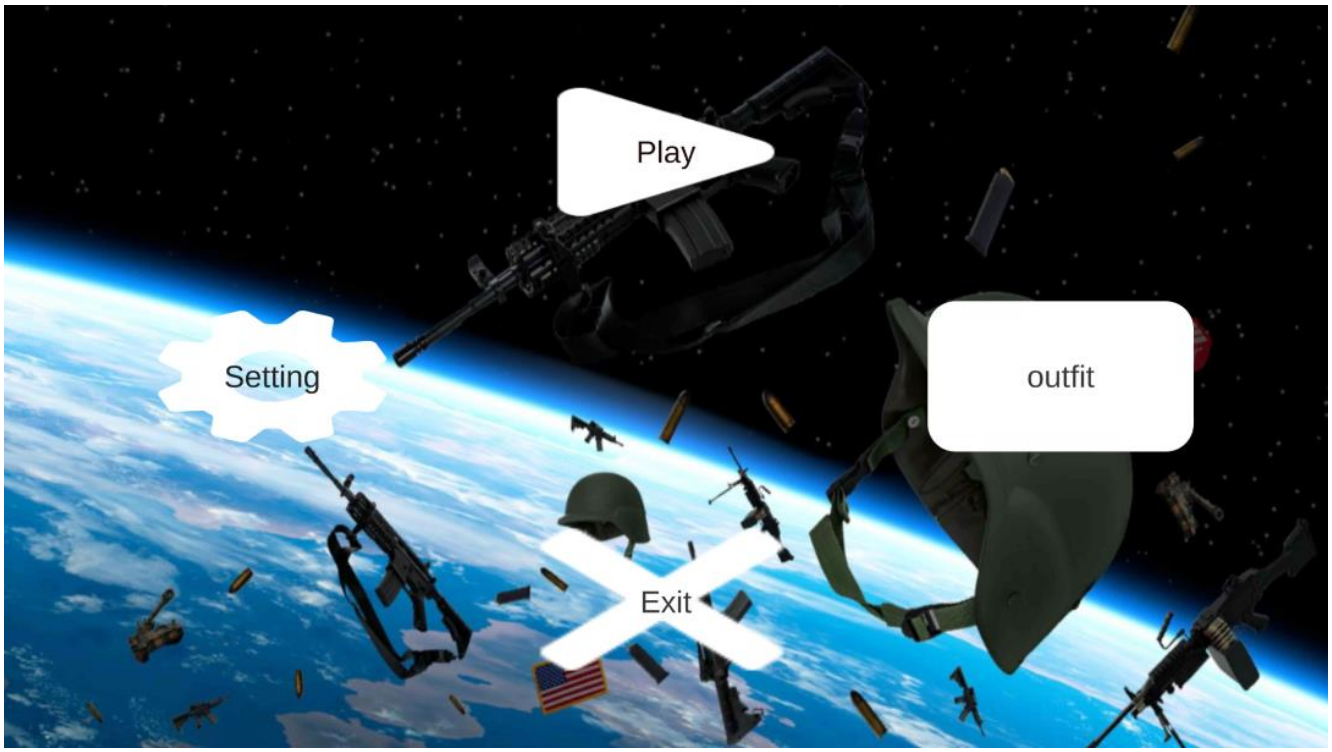


Рис 3.5. Головне меню

3.4.3. Reload

Клас Reload відповідає за перезарядку зброї. Він містить змінні для доступу до кнопки перезарядки, аніматора, який відповідає за відтворення анімації перезарядки, зброї, яка потребує перезарядки, та часу, який необхідний для перезарядки. У методі Start() додається слухач подій для кнопки перезарядки. Метод ReloadAmmo() викликається при натисканні кнопки перезарядки та починає відтворення анімації перезарядки за допомогою змінної _animator. Після закінчення перезарядки викликається метод FinishReload(), в якому зброя перезаряджається за допомогою методу Reload() змінної _weapon, аніматор закінчує відтворення анімації перезарядки, встановлюючи змінну "Reload" у false.

3.4.4. LevelsView

Клас LevelsView відповідає за відображення рівнів гри та їх доступність. У класі використовується константа CurrentLevelKey для збереження поточного рівня гравця в Playerprefs. У класі оголошені поля для представлення префабу для відображення елементів рівня, місця, де вони будуть відображатися, список

дескрипторів рівнів та список представлень рівнів. Під час старту гри викликається метод LoadCurrentLevel() для завантаження поточного рівня гравця та метод GenerateLevelItems() для створення елементів рівня. Метод LoadCurrentLevel() перевіряє, чи існує збереження поточного рівня в Playerprefs. Якщо ні, то поточний рівень встановлюється на 0 та зберігається в Playerprefs. Якщо так, то поточний рівень завантажується з Playerprefs. Метод OnButtonHend() викликається при натисканні на кнопку елемента рівня і запускає відповідний рівень, якщо він доступний для гравця. Метод SetWeaponStatus() використовується для налаштування доступності зброї. Метод GenerateLevelItems() створює елементи рівня з використанням префабу _levelItemViewPrefab та розміщує їх на місці, вказаному в _transformBox. Для кожного елементу рівня встановлюється його номер та статус доступності, а також додається в список _itemLevels. Якщо рівень доступний, встановлюється обробник події натискання кнопки. Після закриття програми або знищення класу збирається список обробників подій на кнопках елементів рівня. Приклад зображено (рис. 3.6.)

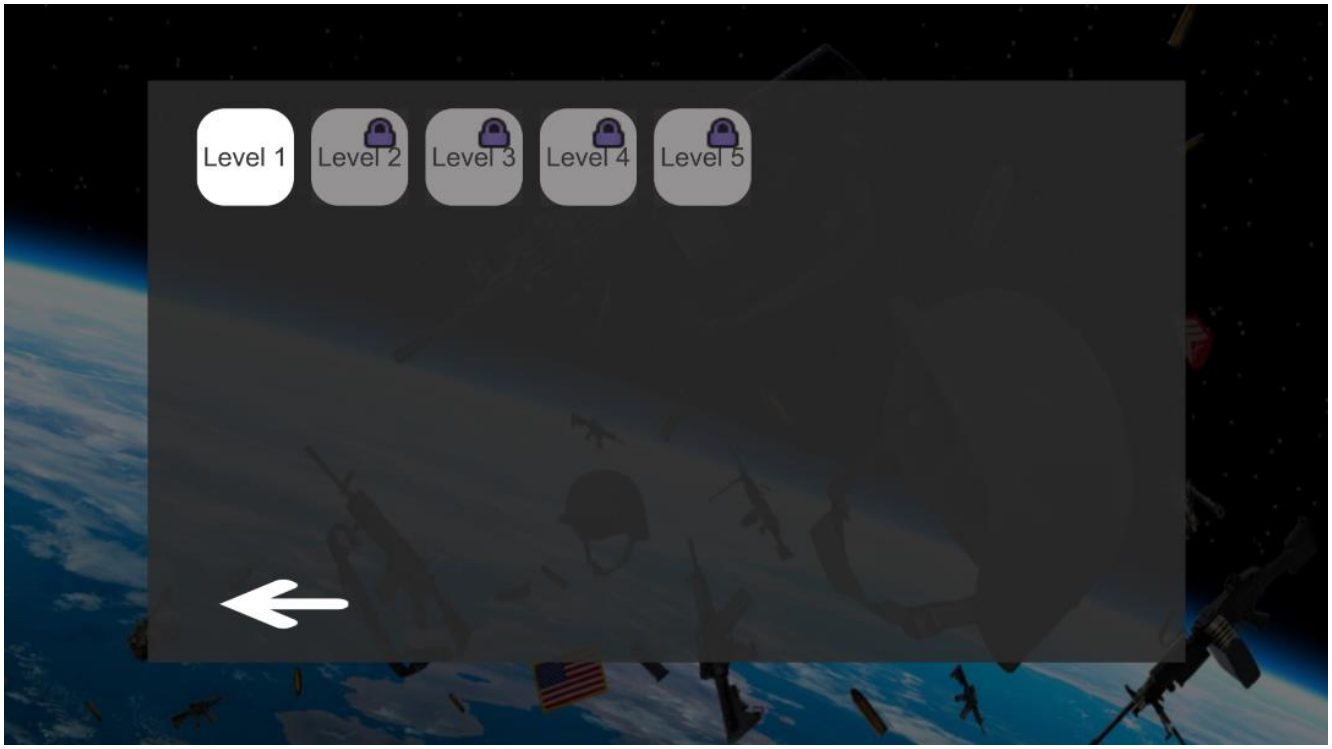


Рис 3.6. Вибір рівня і віткриваються після проходження попереднього

3.4.5. LevelItemDescriptor

Цей клас містить два публічних поля: levelNumber та sceneName. levelNumber представляє номер рівня, а sceneName - назву сцени, яка пов'язана з цим рівнем. Клас оголошений з модифікатором internal, що означає, що він доступний тільки всередині того ж проекту, де він оголошений.

3.4.6. LevelItemView

Клас LevelItemView відповідає за відображення одного елементу у списку рівнів гри. У ньому оголошені змінні для доступу до текстового поля, зображення, яке перекриває кнопку, та самої кнопки. Метод SetLevelNumber встановлює номер рівня для відображення в текстовому полі. Метод SetLockStatus встановлює стан зображення, яке перекриває кнопку. Методи Start та OnDestroy відповідають за підписку та відписку від події натискання на кнопку. Подія onButton спрацьовує при натисканні на кнопку та передає об'єкт LevelItemView в якості параметру, який буде оброблений в батьківському класі.

3.4.7. WeaponList

Клас WeaponList відповідає за відображення списку зброї в грі. У ньому оголошені змінні для доступу до представлення зброї, контейнера, у який будуть додаватися представлення зброї, списку дескрипторів зброї, списку представлення зброї та поточного рівня гри. Клас також містить константи для зберігання ключів, за якими зберігається інформація про вибрану зброю та поточний рівень гри.

Метод Start викликається при запуску сцени і виконує методи для завантаження поточного рівня гри та генерації елементів списку зброї. Метод OnDestroy викликається при знищенні об'єкту та відписує всі обробники подій натискання на кнопки.

Метод LoadCurrentWeapon перевіряє, чи збережена інформація про поточний рівень гри в локальному сховищі. Якщо ні, то встановлюється значення за замовчуванням (0). Якщо так, то метод зчитує збережене значення та виводить його в консоль.

Метод OnButtonHend обробляє натискання на кнопку вибору зброї. Він визначає індекс натиснутого елементу списку та зберігає цей індекс у локальному сховищі за ключем SelectedWeaponKey.

Метод SetWeaponStatus встановлює стан зброї в грі (активована чи ні). Метод GenerateWeaponItems генерує елементи списку зброї на основі переданих дескрипторів зброї. Він створює нове представлення зброї з використанням префабу _weaponsItemView, встановлює назву зброї та її зображення, а також стан замка, що залежить від поточного рівня гри. Якщо зброя доступна, обробник натискання на кнопку додається до списку. У кінці метод додає нове представлення зброї до списку. Приклад зображено (рис. 3.7.)

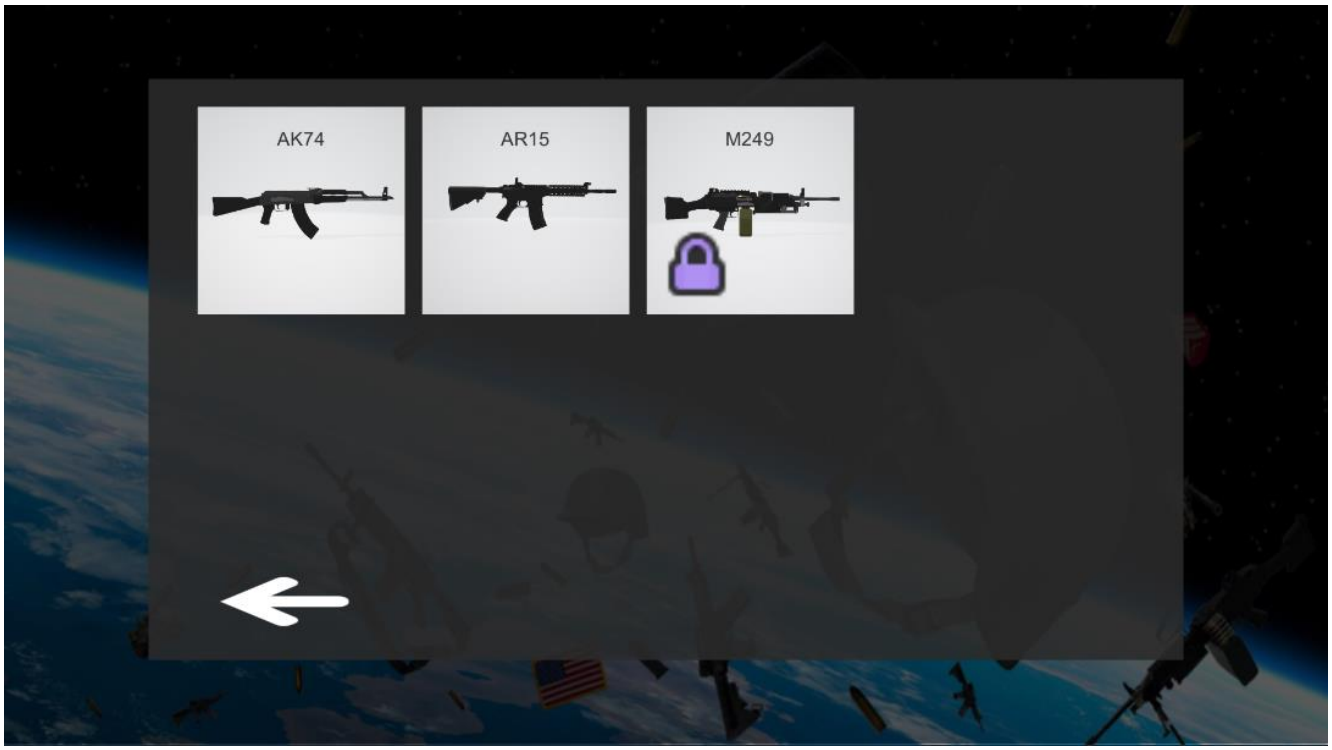


Рис 3.7. Вибір зброї яка відкривається після проходження рівнів

3.4.8. WeaponItemDescriptor

Клас WeaponItemDescriptor містить опис окремого предмета зброї в грі. Він має дві публічні властивості: Name, яка містить назву зброї, і Image, яка містить зображення зброї у вигляді Sprite. Ці властивості дозволяють іншим класам використовувати інформацію про зброю в грі, таку як ім'я та зображення, для

відображення цієї інформації в інтерфейсі користувача або для обчислень, пов'язаних з зброєю. Клас є внутрішнім (internal), що означає, що він доступний лише всередині цього проекту і не може бути використаний в інших проектах.

3.4.9. WeaponsItemView

Клас WeaponsItemView відповідає за відображення одного елементу у списку зброї. У ньому оголошені змінні для доступу до текстового поля з назвою зброї, зображення, яке перекриває кнопку, та самої кнопки. Метод SetWeaponName встановлює назву зброї для відображення в текстовому полі. Метод SetLockStatus встановлює стан зображення, яке перекриває кнопку. Метод SetImage встановлює зображення зброї.

Також в класі оголошується подія onButton, яка спрацьовує при натисканні на кнопку та передає об'єкт WeaponsItemView в якості параметру, який буде оброблений в батьківському класі.

Метод Start підписується на подію натискання на кнопку _button і викликає обробник onButton в разі спрацювання цієї події. Метод OnDestroy відписується від цієї події.

3.4.10. WeaponManager

Клас WeaponManager відповідає за управлінням зброєю гравця. У ньому оголошується константа SelectedWeaponKey, яка містить ключ для зберігання вибраної зброї в PlayerPrefs. Також в класі оголошується масив GameObject[] _weapons, який містить усі доступні зброї гравця.

Змінна _selectedWeaponIndex містить індекс вибраної зброї, початкове значення якої дорівнює 0. Метод Start перевіряє, чи є в PlayerPrefs збережений індекс вибраної зброї, і якщо є, то встановлює його як _selectedWeaponIndex та викликає метод SelectWeapon з цим індексом.

Метод OnDestroy зберігає _selectedWeaponIndex в PlayerPrefs.

Метод SelectWeapon вибирає зброю з масиву _weapons з індексом index і встановлює цю зброю активною, а всі інші - неактивними. Якщо index невірний,

метод повертається. Після вибору зброї _selectedWeaponIndex оновлюється з новим індексом.

Отже, клас WeaponManager забезпечує вибір зброї гравця з масиву доступних зброїв, зберігання індексу вибраної зброї в PlayerPrefs і відповідає за відображення правильної зброї на сцені.

3.4.11. ZombieCounter

Клас ZombieCounter відповідає за лічильник зомбів, які потрібно уничтожити, щоб завершити рівень. У класі оголошена константа CurrentLevelKey, що використовується для збереження номера поточного рівня у PlayerPrefs. Також є змінна zombieCount, яка визначає кількість зомбів, яких потрібно знищити для завершення рівня, і змінна currentZombieCount, яка лічить кількість знищених зомбів.

У методі Start підписуємося на подію уничтоження зомбів SystemHP.OnZombieDestroyed, і у методі OnDestroy відписуємося від неї. Метод IncrementZombieCount викликається при кожному уничтоженні зомбів. Він інкрементує лічильник знищених зомбів currentZombieCount і перевіряє, чи були уничтожені всі зомбі. Якщо так, то завершуємо рівень, виконуючи певні дії (наприклад, переходимо на головне меню гри або на наступний рівень). У даному випадку, відбувається перехід на перший рівень і збільшується лічильник рівнів в PlayerPrefs.

3.5. Опис пул об’єктів

3.5.1. IPoolable

Інтерфейс IPoolable містить оголошення властивостей Transform та GameObject, які повертають зображення трансформації та грифа об'єкта пулу. Інтерфейс також містить подію OnReturnToPool, яка буде викликана, коли об'єкт буде повернений до пулу. Класи, що реалізовують інтерфейс IPoolable, повинні

також реалізувати метод Reset (), який повинен скидати стан об'єкта до початкового стану після повернення його до пулу.

3.5.2. ObjectPool

Даний клас ObjectPool використовується для керування об'єктами пулу в грі. Він містить приватне поле instance, яке є статичним і повертає єдиний екземпляр класу через властивість Instance, щоб запобігти створенню більше одного екземпляру класу.

Також в класі є приватні поля: _activePoolTasks, що є словником з ключем у вигляді типу MonoBehaviour та значенням типу PoolTask, який містить об'єкти пулу, що були активовані, та _objectPoolTransform, що містить трансформацію об'єкта пулу, до якого додаються контейнери для об'єктів пулу.

Конструктор класу встановлює початкові значення для приватних полів _activePoolTasks та _objectPoolTransform.

Метод AddTaskToPool додає новий контейнер для об'єктів пулу до _objectPoolTransform та створює новий PoolTask для цього контейнера, додаючи його до _activePoolTasks.

Метод GetObject повертає об'єкт пулу, який відповідає переданому параметру prefab. Якщо відповідний PoolTask ще не створено, то він створюється за допомогою методу AddTaskToPool. Потім метод GetFreeObject викликається для цього PoolTask, щоб повернути безкоштовний об'єкт пулу.

Метод DisposeTask викликає метод ClearPool для кожного елемента _activePoolTasks, який очищає контейнер пулу та повертає всі його об'єкти до пулу. Цей метод може використовуватися для очищення всього пулу при завершенні гри або зміні сцени.

3.5.3. PoolTask

Пул об'єктів - це механізм, що дозволяє підтримувати пул об'єктів у пам'яті, щоб уникнути витрат на створення нових об'єктів під час виконання програми. Він

містить три поля: "_freeObjects" - список вільних об'єктів, "_objectsInUse" - список об'єктів, які вже використовуються та "_container" - контейнер для зберігання об'єктів.

Конструктор класу ініціалізує ці поля, приймаючи як аргумент змінну "transform", яка представляє контейнер для зберігання об'єктів.

Клас містить метод "GetFreeObject", який приймає параметр "prefab" - це об'єкт, який буде використовуватися для створення нових об'єктів. Метод шукає вільний об'єкт в "_freeObjects". Якщо він знайдений, то він береться зі списку вільних об'єктів, активується та повертається методом. Якщо ж вільного об'єкта немає, то створюється новий об'єкт з використанням "Instantiate" та повертається методом. Кожен створений об'єкт додається до списку "_objectsInUse".

Метод "ReturnToPool" додає повернений об'єкт до списку "_freeObjects", видаляє його зі списку "_objectsInUse", відписує його від події "OnReturnToPool" та виключає його, щоб звільнити пам'ять. Об'єкт також додається до контейнера "_container".

Метод "ReturnAllToPool" повертає всі об'єкти зі списку "_objectsInUse" до "_freeObjects".

Метод "ClearPool" видаляє всі об'єкти зі списку "_objectsInUse" та "_freeObjects", щоб звільнити пам'ять.

3.6. Опис класів зомбі

3.6.1. MonsterBehaviourSwitcher

Клас "MonsterBehaviourSwitcher", який наслідується від "MonoBehaviour" і реалізує поведінку монстра у грі. У класі є змінна "zombieSpeed", яка визначає швидкість монстра. Також є змінні "_animator", "_patrolArea", "_followPlayer" і "_detectionZone", які зберігають посилання на аніматор монстра, область патрулювання, скрипт переслідування гравця та зону детекції відповідно.

У методі "Start" встановлюється початковий стан монстра, де він патрулює. У методі "OnTriggerEnter" перевіряється, чи знаходиться гравець у зоні детекції,

якщо так, то монстр починає переслідувати гравця. У методі "OnTriggerExit" перевіряється, чи вийшов гравець з зони детекції, якщо так, то монстр знову починає патрулювати.

Під час зміни стану монстра, змінюється значення змінної "_isPlayerDetected", а також встановлюється відповідний стан області патрулювання і скрипту переслідування гравця. Залежно від стану монстра, також змінюється його швидкість, яку встановлюємо через змінну "zombieSpeed".

3.6.2. PatrolArea

Клас "PatrolArea", який наслідується від "MonoBehaviour" і реалізує патрулювання монстра у грі. У класі є змінні "_waypoints" і "speed", які зберігають координати точок маршруту монстра та його швидкість відповідно.

У методі "Start" встановлюється початкова точка маршруту.

У методі "Update" монстр рухається в напрямку наступної точки маршруту, обертаючись в цей бік. Монстр рухається зі швидкістю, вказаною в змінній "speed". Якщо монстр досягає близької відстані до точки маршруту, то встановлюється наступна точка як активна. Якщо монстр досягає останньої точки маршруту, то активною стає перша точка маршруту. Приклад зображено (рис. 3.8.)

Рис 3.8. Патрулювання зомбі

3.6.3. FollowPlayer

Клас "FollowPlayer" унаслідується від класу "MonoBehaviour", що забезпечує можливість використання методів Unity для керування об'єктами в грі.

У цьому класі є два змінні, які можна змінювати в інспекторі Unity: "_target" (ціль, за якою буде слідувати об'єкт) і "speed" (швидкість руху об'єкта).

У методі "Update" обчислюється вектор напрямку від поточної позиції об'єкта до цілі, нормалізується і потім використовується для налаштування позиції та орієнтації об'єкта.

Спочатку об'єкт повертається в напрямку цілі за допомогою методу "LookAt". Потім вектор напрямку нормалізується, щоб забезпечити рух в одному напрямку з фіксованою швидкістю "speed". Нарешті, об'єкт рухається в передньому напрямку шляхом додавання до поточної позиції вектору "transform.forward", помноженого на добуток швидкості на час "speed * Time.deltaTime". Приклад зображено (рис. 3.9.)

Рис 3.9. Зомбі біжить за гравцем

3.6.4. ZombieAttack

Клас "ZombieAttack" містить змінні, які можна змінювати в інспекторі Unity: "_projectilePrefab" (префаб снаряду), "_projectileSpawnPoint" (точка спавна снаряду), "_animator" (об'єкт аніматора), "_playerTransform" (трансформація головного героя гри), "_attackDistance" (відстань атаки зомбі) і "attackDelay" (затримка між атаками).

У методі "Update" визначається, чи знаходиться головний герой на відстані, достатній для атаки зомбі. Якщо відстань менша або рівна встановленому значенню "_attackDistance", і пройшов достатній час з моменту останньої атаки (визначається затримкою "attackDelay"), то зомбі виконує атаку.

У методі "Attack" створюється снаряд з префаба "_projectilePrefab" і розміщується в точці "_projectileSpawnPoint". Снаряд знищується після двох секунд використовуючи метод "Destroy". Крім того, викликається тригер анімації "Attack" на об'єкті "_animator", який відповідає за відтворення анімації атаки зомбі. Приклад зображено (рис. 3.10.)

Рис 3.10. Зомбі атакує гравця

3.6.5. ZombieDamage

Клас "ZombieDamage" містить змінні, які можна змінювати в інспекторі Unity: "damage" (кількість шкоди, яку завдає зомбі), "damageTag" (тег об'єкта, на який зомбі може нанести шкоду) і "fireDelay" (затримка між атаками).

У методі "Update" змінюється час, який пройшов з моменту останньої атаки зомбі.

У методі "OnTriggerStay" перевіряється тег об'єкта, на який потрапив колайдер зомбі. Якщо тег співпадає з встановленим значенням "damageTag", і пройшов достатній час з моменту останньої атаки (визначається затримкою "fireDelay"), то зомбі завдає шкоди головному герою гри.

Для цього отримуємо компонент здоров'я головного героя методом "GetComponent<SystemHP>()", якщо він є. Потім викликаємо на ньому метод "TakeHit", який відповідає за зменшення здоров'я на кількість, яка вказана в змінній "damage". Після завершення атаки збройовик чекає, поки не пройде затримка "fireDelay", перш ніж здійснити наступну атаку.

3.7. Тестування

Гру було протестовано за основним сценарієм такими як:

1. Меню:
- Перевірка, чи можливо відкрити меню зброї та вибору рівня.
- Перевірка, чи заблоковані всі рівні, крім початкового.
- Перевірка, чи відкривається наступний рівень після успішного завершення поточного рівня.
2. Логіка рівня:
- Перевірка, чи виконуються умови для перемоги на кожному рівні.
- Перевірка, чи спрацьовує поразка, коли здоров'я гравця падає до 0.
- Перевірка, чи відкривається наступний рівень після успішного завершення поточного рівня.
3. Логіка стрільби:
- Перевірка, чи відбувається стрільба, коли гравець націлюється на ворога.
- Перевірка, чи зупиняється стрільба, коли набої закінчуються.
- Перевірка, чи можливо перезарядити зброю після закінчення набоїв.
- Перевірка, чи можливо продовжити стрільбу після перезарядки зброї.
4. Логіка монстрів:
- Перевірка, чи монстри патрулюють зону та рухаються по встановленій траєкторії.
- Перевірка, чи монстри розпочинають переслідування та атакують гравця, коли він потрапляє в їх зону.
- Перевірка, чи зомбі в рівнях, де потрібно протриматись певний час, йдуть безпосередньо на гравця.
5. Логіка атаки монстра:
- Перевірка, чи монстри атакують гравця, коли він перебуває в їх зоні атаки.
- Перевірка, чи спрацьовує урон

Таблиця 3.1 Тест кейси

Тест-кейс	Опис кейсу	Очікуваний результат	Фактичний результат	Відмітка
1	Запуск гри	Гра запускається без помилок	Працює згідно очікуваному результату	+
2	Меню:	Гравець може обрати відкриту зброю та рівень	Працює згідно очікуваному результату	+
3	Логіка рівня	Гравець може пройти рівинь виконавши певні умови. Або буде поразка, коли хп гравця впаде до 0	Працює згідно очікуваному результату	+
4	Логіка стрільби	При наведені зброї на зону монстра йде стрільба, й поки є набої, а коли скінчаться потрібно перезарядити зброю	Працює згідно очікуваному результату	+

Продовження таблиці 3.1 Тест кейси

Тест-кейс	**Опис кейсу**	**Очікуваний результат**	**Фактичний результат**	**Відмітка**
5	Логіка монстрів	Патрулює, коли в його зону входить гравець, то монстр біжить за гравцем	Працює згідно очікуваному результату	+
6	Логіка атаки монстра	Атакує, коли добіг до гравця.	Працює згідно очікуваному результату	+

Висновки до розділу 3

Було виконано наступні пункити:

Структура файлів гри: Успішно створено структуру файлів гри, що організовує різні компоненти, такі як файли з кодом, графічні ресурси, звукові ефекти та інші ресурси. Це допоможе зберегти проект організованим і зробить його легше розширюваним у майбутньому.

Опис логіки гри: Ретельно описано логіку гри, що включає правила, механіку взаємодії гравця з оточуючим світом, умови перемоги або поразки, систему очків та інші важливі аспекти. Це надає зрозумілу основу для реалізації гри.

Логіка меню: Розроблено логіку меню, яка дозволяє гравцю навігувати та налаштовувати гру. Це може включати початок гри, налаштування графіки або звуку, вибір рівнів тощо.

Логіка рівня: Реалізовано логіку рівнів гри, що включає генерацію різних етапів або пройдених рівнів, розміщення об'єктів на рівні, обробку взаємодії гравця з ними та умови для завершення рівня.

Логіка стрільби: Виконано розробку логіки стрільби, яка включає у себе обробку вибору цілей, керування кутом та силою пострілу, обробку зіткнень та вплив стріл на об'єкти у грі.

Логіка монстрів: Реалізовано логіку монстрів, яка включає їх поведінку, рух, розміщення на рівні та способи взаємодії з гравцем.

Логіка атаки монстра: Була розроблена логіка атаки монстра, включаючи його методи атаки, поведінку при підході до гравця, можливість уникати атак та інші аспекти, пов'язані з взаємодією монстрів з гравцем.

Тестування: Важливо провести тестування гри, щоб переконатися, що всі аспекти логіки гри, включаючи меню, рівні, стрільбу, монстрів та атаки монстра, працюють належним чином. Тестування допоможе виявити й виправити можливні помилки та проблеми зі стабільністю та функціональністю гри.

Тест-кейси: Для ефективного тестування варто розробити тест-кейси, які включатимуть різні сценарії тестування для перевірки кожного аспекту гри. Тест-кейси можуть містити вхідні дані, кроки виконання тесту та очікувані результати. Вони сприятимуть систематичному підходу до тестування та допоможуть забезпечити високу якість гри.

ВИСНОВКИ

Проаналізовано популярні жанри відеоігор та обрано жанр "Шутер".

Розглянуто актуальність ігор, зокрема хто в них грає та їх прибуток.

Проведено аналіз подібних ігрових додатків, визначено їхні сильні та слабкі сторони, успішність та оригінальні рішення. Обрано, що саме буде в ігровому додатку.

Проведено аналіз середовищ розробки для відеоігор. Проаналізовано їхні можливості, обмеження та придатність для проекту. Обрано оптимальне середовище розробки, яке найкраще підходить.

Розроблено ігровий додаток з різноманіттям рівнів.

Проведено тестування гри, після якого було виявлено недоліки та помилки, які були усунені.

ПЕРЕЛІК ПОСИЛАНЬ

1. **Video game**

https://en.wikipedia.org/wiki/Video_game

2. **Who Is Playing Games?**

https://www.pewresearch.org/internet/2008/09/16/part-1-1-who-is-playing-games/

3.Best Game Engines for Game Development in 2023

https://www.juegostudio.com/blog/best-game-engines

4.WHY IS THE UNITY GAME ENGINE CONSIDERED THE BEST? PROS AND CONS

https://stepico.com/blog/why-is-unity-the-best-game-engine-pros-and-cons/

5.The Pros & Cons of Creating 3D Content With Blender Software

https://www.epidemicsound.com/blog/blender-software/

6.Why Gamers Can’t Stop Playing First-Person Shooters

https://www.newyorker.com/tech/annals-of-technology/why-gamers-cant-stop-playing-first-person-shooters

7. Why Top-Down?

https://www.instantkingdom.com/2010/09/why-top-down

8.Game Development using Unity Game Engine

https://ieeexplore.ieee.org/document/10007155

9.Discussion on Educational Games Based on Unity

https://dl.acm.org/doi/10.1145/3578837.3578847

10.Design and Analysis of Clothing Catwalks Taking into Account Unity's Immersive Virtual Reality in an Artificial Intelligence Environment

https://pubmed.ncbi.nlm.nih.gov/35479606/

11.Programming C# 10

https://books.google.com.ua/books?hl=uk&lr=&id=0HJ_EAAAQBAJ&oi=fnd&pg=PT21&dq=C%23+programming+languages&ots=7-Z599opNT&sig=Ih2xuxILdJzyCvISJiRrU2DmWQ4&redir_esc=y#v=onepage&q=C%23%20programming%20languages&f=false

12. APLIKASI GAME FIRST PERSONAL SHOOTER (FPS) BERBASIS ANDROID
https://jurnal.unived.ac.id/index.php/jmi/article/view/3685

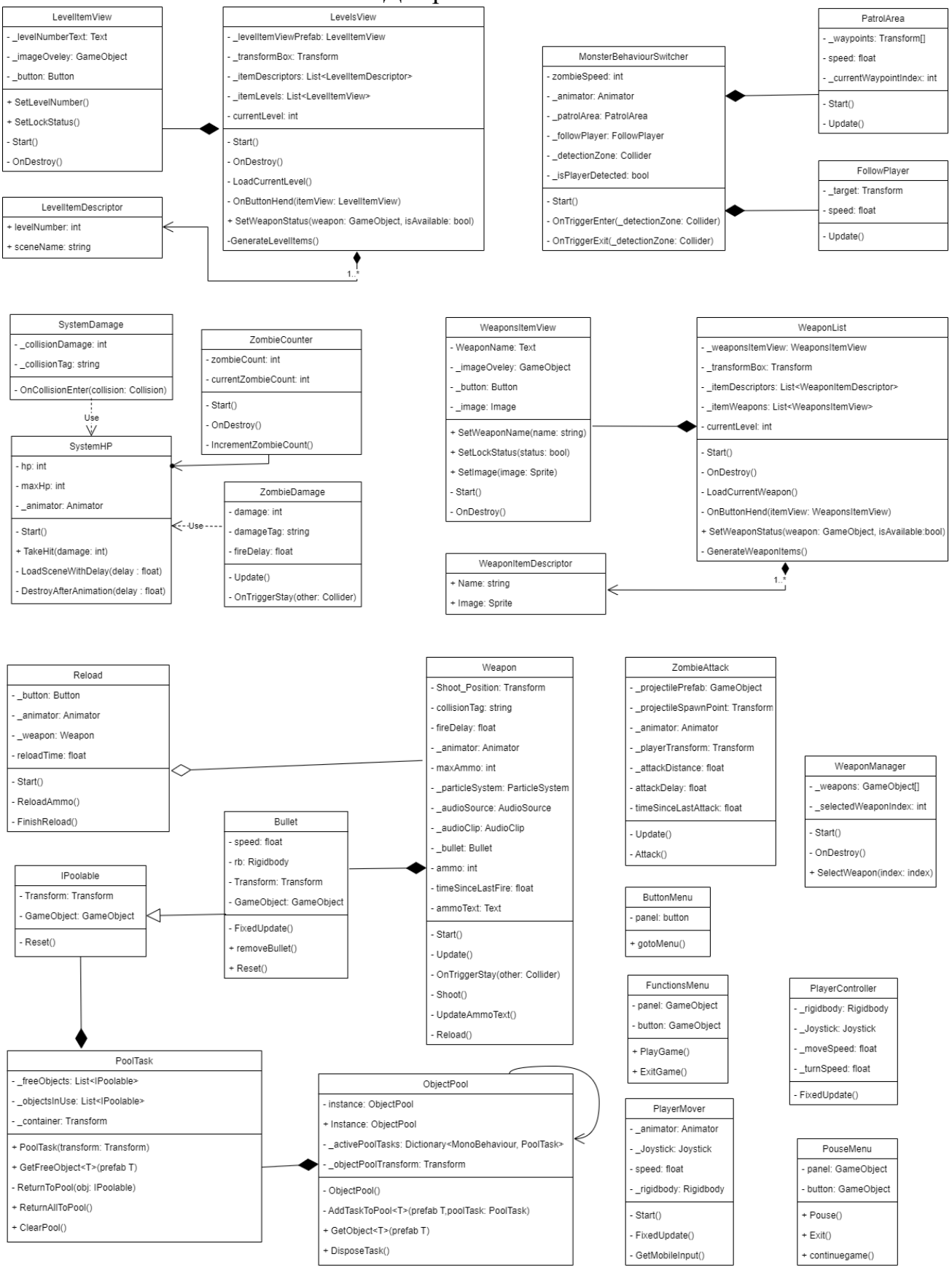
13. **The metaverse, but not the way you think: game engines and automation beyond game development**
https://www.tandfonline.com/doi/full/10.1080/15295036.2022.2080850

14.From Asymptomatics to Zombies:
https://dl.acm.org/doi/abs/10.1145/3544548.3581573

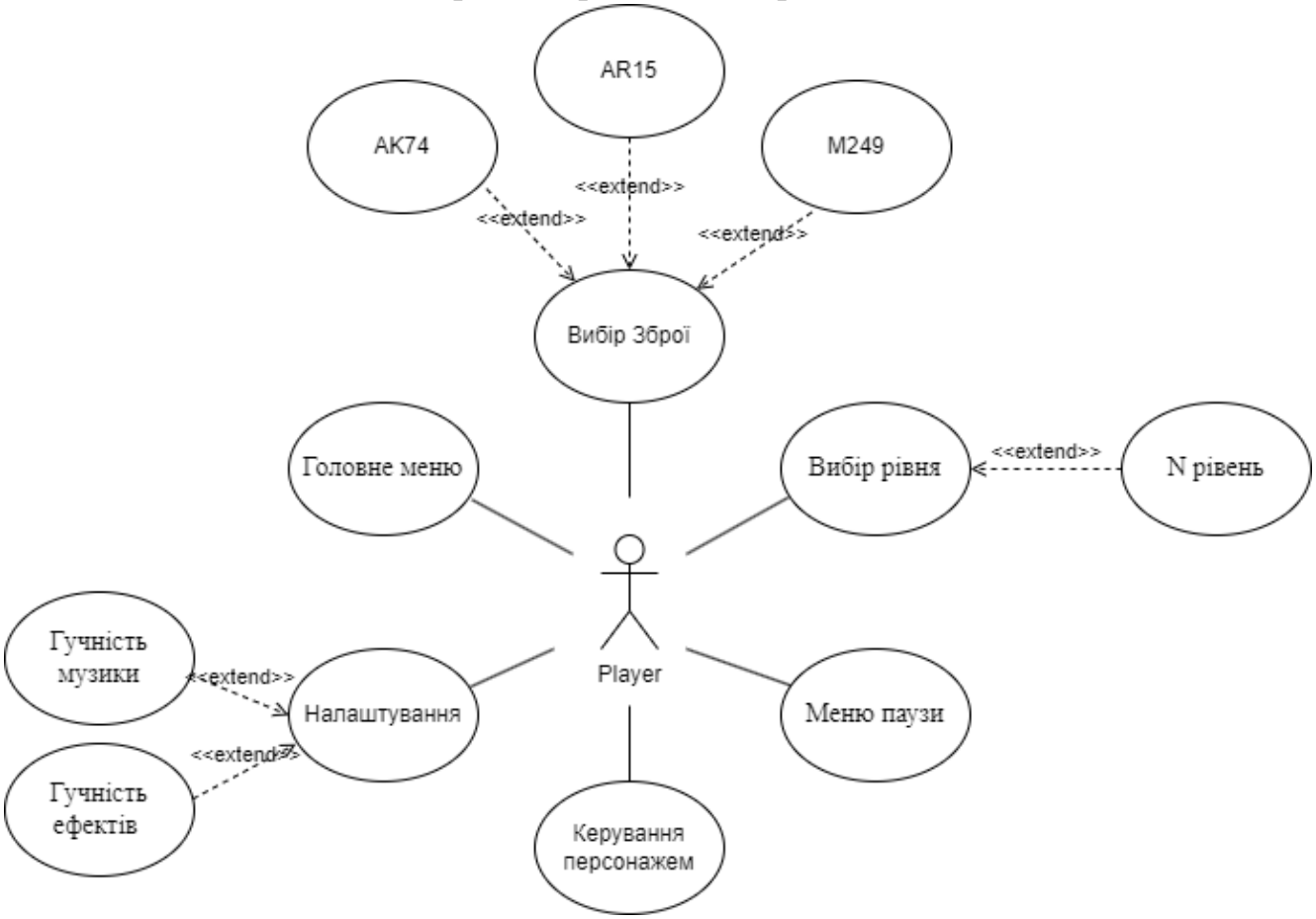
15.Large Estimate Variations in Assessed Energy Expenditure and Physical Activity Levels
https://www.mdpi.com/1660-4601/20/2/1548

ДОДАТОК А

Діаграма класів



Діаграма Варіантів Використання



ДОДАТОК Б

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка гри “military quest ” під платформу Android у жанрі “"шутер 2.5d"” з використанням ігрового рушія Unity мовою C#

Виконав студент 4 курсу
Групи ПД-42
Олексенко Роман Григорович
Керівник роботи
Доктор філософії
Дібрівний Олесь Андрійович

Київ – 2023

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи** – підвищення зацікавленості користувача в грі military quest в жанрі “шутер 2.5d” за рахунок реалізації різноманіття рівнів.
- **Об’єкт дослідження** – ігровий процес у грі в жанрі "shooter".
- **Предмет дослідження** – інструменти створення ігрового додатку в жанрі "shooter" у 2.5D просторі.

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Провести аналіз та обрати жанр відеогри.
2. Розглянути актуальність ігор.
3. Провести аналіз подібних ігрових додатків для усунення недоліків в своєму додатку.
4. Провести аналіз та обрати середовища для розробки гри.
5. Розробити ігровий додаток.
6. Провести тестування відеогри та усунення помилок.

АНАЛІЗ АНАЛОГІВ

Показник	Dead Nation	Zombie Anarchy	Killing Floor: Calamity	Military Quest
Платформи	PlayStation, Android	iOS, Android та Windows 10	Android	Android
Безкоштовність продукту	-	+	-	+
Вибір зброї	+	-	+	+
Наявність додаткових ефектів під час взаємодії	+	+	+	-
Вимогливість	-	-	-	+
Мультиплеєр	+	+	+	-
Різноманітність рівнів	-	+	+	+
Реклама в грі	+	-	+	+

4

ВИМОГИ ДО ІГРОВОГО ДОДАТКУ

1.Функціонал головного меню.
2.Функціонал гравця.
3.Функціонал зомбі.
4.Функціонал по зброї та її вибору.
5.Функціонал меню паузи.
6.Функціонал стрільби.
7.Функціонал здоров’я та нанесення урона
8.Різноманітні рівні та функціонал їх проходження

5

КОНЦЕПТ ГРИ

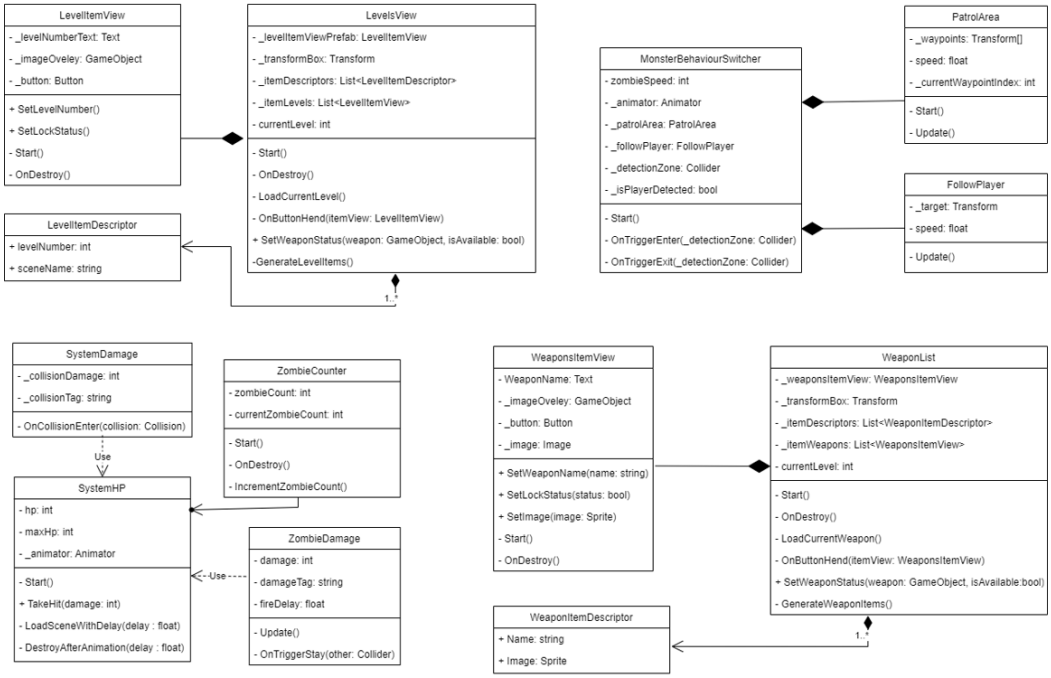
1.Основним в грі є різноманітність рівнів які потрібно пройти. Після проходження рівня відкривається наступний рівень і зброя.
2.Стрільба при наведенні на монстра та його знищення.
3.Монстри патрулюють та атакують гравця.
4.В грі можна проходити рівні.
5.В рі можна оберати зброю.

6

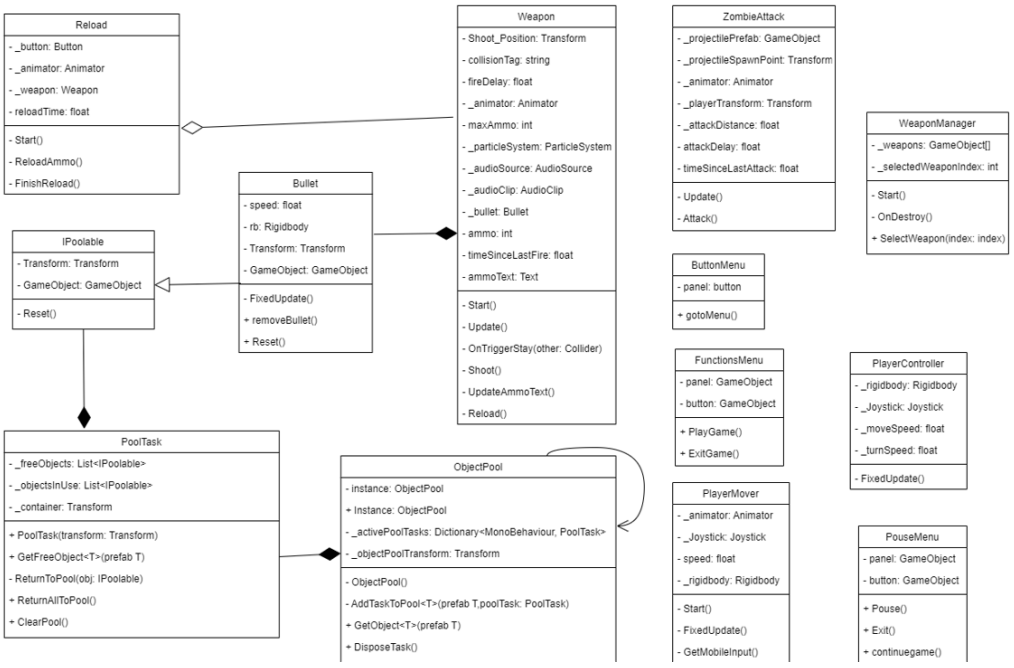
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

7

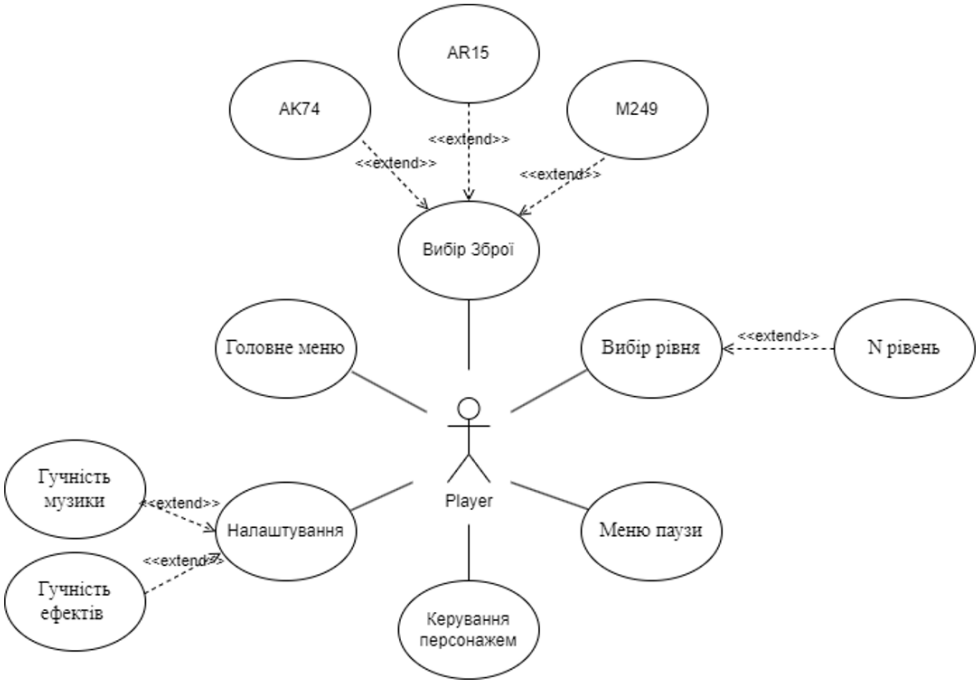
МЕТОДИ ТА КЛАСИ ПРОГРАМИ



МЕТОДИ ТА КЛАСИ ПРОГРАМИ

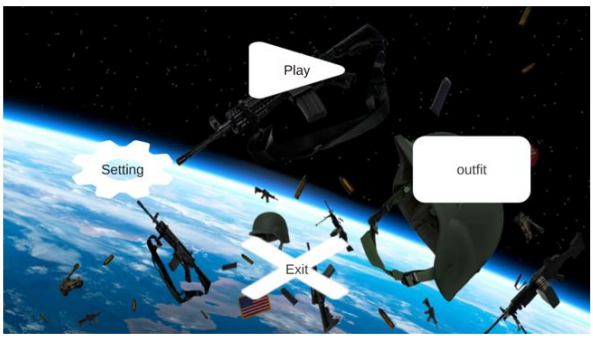


ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ



10

ЕКРАННІ ФОРМИ



Головне меню гри

Ігровий процес

11

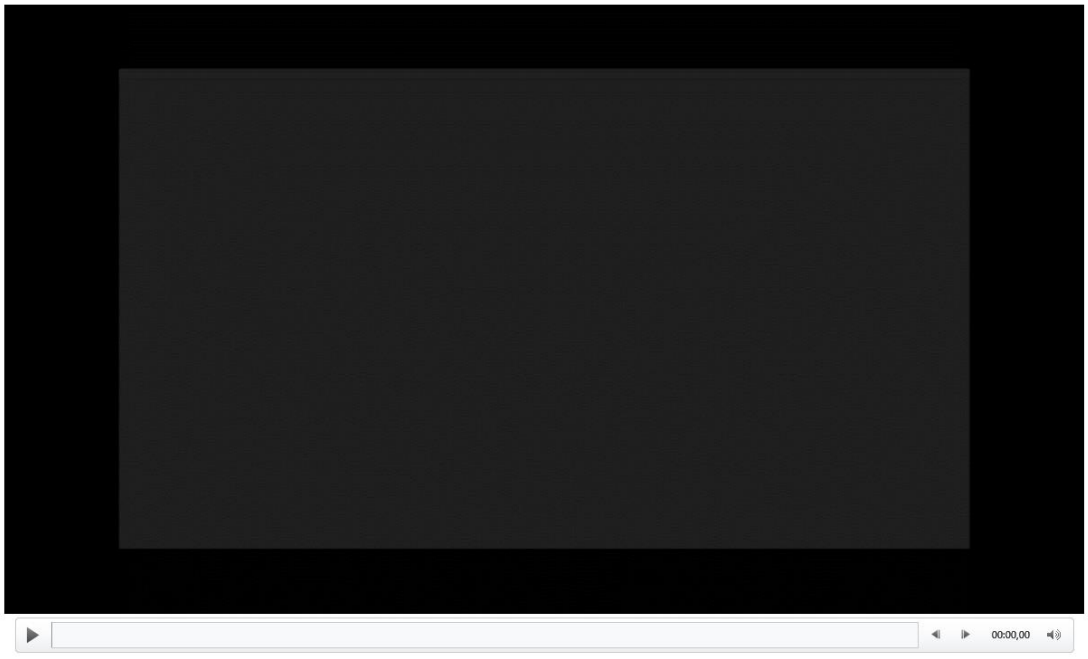
АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1.Олексенко Р.Г. Дослідження відеоігор в жанрі “Шутер”/ Матеріали всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інфокомунікаційних технологіях». 20.04.2023, ДУТ, м.Київ 2023 с. 131-132.

2.Олексенко Р.Г. Сучасні інформаційні технології в україні і світі/ IV Всеукраїнська науково-технічна конференція «Сучасний стан та перспективи розвитку IOT ». Збірник тез.05.04.2023, ДУТ, м.Київ 2023 с. 31-32.

12

ВІДЕО ПРЕДСТАВЛЕННЯ ГРИ



13

ВИСНОВКИ

1.Проаналізовано популярні жанри відеоігор та обрано жанр "Шутер".

2.Розглянуто актуальність ігор, зокрема хто в них грає та їх прибуток.

3.Проведено аналіз подібних ігрових додатків, визначено їхні сильні та слабкі сторони, успішність та оригінальні рішення. Обрано, що саме буде в ігровому додатку.

4.Проведено аналіз середовищ розробки для відеоігор. Проаналізовано їхні можливості, обмеження та придатність для проекту. Обрано оптимальне середовище розробки, яке найкраще підходить.

5.Розроблено ігровий додаток з різноманіттям рівнів.

6.Проведено тестування гри, після якого було виявлено недоліки та помилки, які були усунені.

ДЯКУЮ ЗА УВАГУ!