ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА ГРИ ЖАНРУ ЕКШН-РОГАЛИК “UNDERGROUNDER” ДЛЯ ПЛАТФОРМИ ANDROID НА ДВИГУНІ UNITY**»

Виконав: студент 4 курсу, групи ПД–41
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

Личаний В.А.
(прізвище та ініціали)

Керівник Дібрівний О.А.
(прізвище та ініціали)

Рецензент

(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

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

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____Негоденко О.В.
“____” _____2022 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТА

Личаний Владислав Андрійович
(прізвище, ім’я, по батькові)

1. Тема роботи: «Розробка гри жанру екшн-рогалик “Undergrounder” для платформи Android на двигуні Unity »

Керівник роботи: Дібрівний Олесь Андрійович, доктор філософії
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «16» лютого 2022 року №.

2. Строк подання студентом роботи «6» червня 2022 року

3. Вхідні дані до роботи:

3.1 Науково-технічна література з питань, пов’язаних з програмним забезпеченням щодо розробки ігор.
3.2 Офіційна документація Unity.
3.3 Офіційна документація Microsoft Visual Studio.

4. Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).

4.1 Аналіз актуальності та проблематики розроблюваної гри.
4.2 Аналіз та вибір інструментів для реалізації мобільної гри.
4.3 Проектування мобільної гри.
4.4 Висновки.

5. Перелік демонстраційного матеріалу (назва основних слайдів)

5.1 Титульний слайд.

5.2 Мета, об’єкт, предмет, наукова новизна дослідження.

5.3 Актуальність.

5.4 Аналіз аналогів.

5.5 Технічні завдання.

5.6 Програмні засоби та інструменти реалізації.

5.7 Реалізація програми.

5.8 Апробація результатів дослідження.

5.9 Висновки.

5.10 Кінцевий слайд.

6. Дата видачі завдання «11» квітня 2022р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	11.04.2022	Виконано
2	Дослідження аналогів та актуальності додатку	13.04.2022	Виконано
3	Аналіз та вибір інструментів для розробки додатку	14.04.2022	Виконано
4	Проектування та реалізація	02.05.2022	Виконано
5	Вступ, висновки, реферат	17.05.2022	Виконано
6	Розробка обов‘язкових демонстраційних матеріалів	18.05.2022	Виконано
7	Попередній захист роботи	06.06.2022	
8	Здача роботи	06.06.2022	

Студент _____ Личаний В.А
(підпис) (прізвище та ініціали)

Керівник роботи _____ Дібрівний О.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи 43 с., 1 табл., 23 рис., 15 джерел.

Ключові слова: ігровий двигун, екшн, рогалик, Unity, мобільна гра, відеогра.

Об’єкт дослідження - ігровий процес в жанрі екшн-рогалик.

Предмет дослідження - програмний продукт жанрі екшн-рогалик для мобільної платформи Android.

Мета роботи - Покращення користувацького досвіду із внутрішньожанровим різномаїттям відеогр типу екшн-рогалик для мобільних платформ за рахунок розробки гри із фокусом на близький контакт із ворогами.

Наукова новизна – створення нових та перетворення відомих ігрових механік, притаманних для ігор жанра екшн рогалик; акцент на близької взаємодії персонажа гравця із ігровими ворогами та яскравому дизайні гри, що зачепить гравця.

У дипломному проекті був проведений аналіз ринку мобільних та комп’ютерних ігор жанру екшн рогалик. Проаналізовано та виявлено переваги і недоліки програмних інструментів розробки. Проаналізовано особливості розробки мобільних ігор.

Мобільну гру було створено за допомоги ігрового двигуна Unity. Ігрова логіка написана у середовищі розробки Microsoft Visual Studio Community 2019 на мові програмування C#. В якості мобільної платформи було обрано операційну систему Android. Для створення елементів графічного дизайну та прототипування гри було використано Aseprite.

Дана мобільна гра може слугувати як і швидким способом ненадовго відволіктися, так і повноцінною грою, яка дозволить отримувати задоволення від подолання перешкод.

ЗМІСТ

ВСТУП

Об’єкт дослідження - ігровий процес в жанрі екшн-рогалик

Предмет дослідження - програмний продукт жанрі екшн-рогалик для мобільної платформи Android.

Мета роботи - Покращення користувацького досвіду із внутрішньожанровим різномаїттям відеогр типу екшн-рогалик для мобільних платформ за рахунок розробки гри із фокусом на близький контакт із ворогами.

Наукова новизна – створення нових та перетворення відомих ігрових механік, притаманних для ігор жанра екшн рогалик; акцент на близької взаємодії персонажа гравця із ігровими ворогами та яскравому дизайні гри, що зачепить гравця.

У дипломному проекті був проведений аналіз ринку мобільних та комп’ютерних ігор жанру екшн рогалик. Проаналізовано та виявлено переваги і недоліки програмних інструментів розробки. Проаналізовано особливості розробки мобільних ігор.

Мобільну гру було створено за допомоги ігрового двигуна Unity. Ігрова логіка написана у середовищі розробки Microsoft Visual Studio Community 2019 на мові програмування C#. В якості мобільної платформи було обрано операційну систему Android. Для створення елементів графічного дизайну та прототипування гри було використано Aseprite.

Дана мобільна гра може слугувати як і швидким способом ненадовго відволіктися, так і повноцінною грою, яка дозволить отримувати задоволення від подолання перешкод.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальна характеристика відеоігор

Відеогра - це інтерактивний вид розваг, який є програмним забезпеченням, створеним для комп'ютерів та спеціалізованих платформ, мета якого у взаємодії людини – гравця – з віртуальним простором або іншими гравцями за засобами цього самого віртуального простору.

Перші максимально примітивні відеоігри почали з'являтися у 1960-х роках з поширенням комп'ютерних систем у навчальних та наукових закладах, і не були чимось масовим та вагомим у світі розваг. Пізніше, в 70-х стався сплеск аркадних автоматів - масивних кабін з екранами і клавішами, які відтворювали одну конкретну гру.

Організовували навіть спеціальні зали, в які люди ходили лише щоб пограти у безліч аркадних автоматів. А в 80-х почали набирати популярності домашні ігрові консолі скромних розмірів, які використовують знімні носії для ігор.

І хоча вони сильно відставали від аркадних кабін у плані графіки, потужностей та можливостей на початку свого шляху, під кінець 90-х вони перевершили за популярністю решту відеоігрових платформ. Тоді ж ігри для персональних комп'ютерів наздогнали консольні. Розвиток відеоігор був тісно пов'язаний з технологічним прогресом та платформами, для яких вони вироблялися, і їм довелося пройти великий шлях від одноколірного тенісу “Pong” до сучасних тривимірних кінематографічних комплексних блокбастерів нашого часу.

Рисунок 1.1 - Аркадна кабіна першої комерційно успішної відеоігри Pong

Сьогодні відеоігри створюються для безлічі платформ особистого використання, як для універсальних (комп'ютерів та мобільних телефонів) - так і для спеціалізованих під відеоігри платформ (відеоігрові консолі).

Поширюються та монетизуються ігри за моделлю аналогічною фільмам та музиці: у цифровому форматі та на фізичних носіях, які можна придбати в онлайн та рітейлових магазинах відповідно. Однак з 2010-х років стала все більш поширена сервісна модель, ближча до деяких видів монетизації програмного забезпечення - це моделі використання підписки, коли користувач сплачує невелику суму за використання продукту в рівні часові відрізки (місяць, рік), а також free-to-play, коли продукт є безкоштовним, але користувач може платити за окремі елементи та функціонал усередині ПЗ або ігри.

Відмінність відеоігор від багатьох інших видів медіа безпосередньо у можливості взаємодії гравця на внутрішньоігрові процеси відповідно до внутрішньої логіки. Вони тому і називаються іграми, тому що їх результат залежить від впливу та дій користувача, який сильно чи хоча б мінімально залучений до ігрового процесу. Іншими словами відеоігри інтерактивні. Відеоігри

використовуються для безлічі різних цілей: як і багато інших видів розваги та мистецтва, вони можуть бути варіантом відпочинку або проведенням дозвілля, джерелом багатого спектру емоцій, служити навчальним матеріалом і створювати симуляційні середовища, допомагати з науковими дослідженнями та багато іншого, хоча найпоширенішими є все ж таки розважальні цілі.

Наприклад, у багатьох освітніх системах світу існують окремі або навіть спеціально для них розроблені відеоігри, тому що при такій залученості, яку учням і студентам дарують ігри, матеріал набагато краще засвоюється, також ефект підсилюється візуальною реакцією у відповідь, у учнів, які грають у ігри, більше розвивається фантазія і вони сильніше проявляють зацікавленість у вивченні предмета де були застосовані ігри як інструмент пояснення матеріалу.

Окрему категорію займають симулятори. Для багатьох складних професій, де допуск помилок у реальному житті неприпустимий, створюють спеціальні віртуальні середовища для відточення всіх необхідних навичок. Пілоти, космонавти, хірурги, навіть пожежники та багато інших професій використовують відеоігрові симулятори, які мають найбільш наближені до реальності умови для навчання та тренування особливих умінь, які згодом будуть необхідними для виконання вже справжньої роботи.

Відеоігри також знайшли своє застосування у медицині. Нехай і не так широко, але вони використовуються в реабілітації хворих для розвитку їх фізичних та розумових здібностей. Звичайно, говорячи про відеоігри в медицині, не можна не згадати їхню велику роль у лікуванні багатьох психологічних проблем і захворювань, тому що ігри допомагають справлятися зі стресом, заспокоюють своєю передбачуваністю, і допомагають багатьом людям за рахунок медитативності. Однак все ж таки варто розуміти, що це лише мала частина в процесі лікування, а в деяких випадках вони можуть тільки нашкодити.

Рисунок 1.2 - Відеогра з системою відстеження рухів для фізичних вправ хворих людей

Індустрія відеоігор - це величезний бізнес: розробка сучасних великомасштабних ігор вимагає кількох років виробництва, грошові суми, що обчислюються мільйонами, та сотні фахівців. Прибуток таких відео ігрових компаній як Sony, Tencent, Microsoft, Nintendo, Electronic Arts, Ubisoft та багатьох інших, обчислюється мільярдами доларів у кожній. Але в той же час на ринку час від часу з'являються і набувають величезної популярності відеоігри, створені невеликою командою або навіть однією людиною зі скромним бюджетом або зовсім без нього. Такі проекти називаються "інді" (від англійської "independent"), тобто створені незалежними розробниками.

1.2 Характеристика та особливості жанру екшн рогалик

Як і в інших видів мистецтв та розваг, у відеоігор присутній поділ за жанрами. Жанр - це категорія творів (у даному випадку відеоігор), об'єднаних подібними ознаками і характерними елементами. Серед відеоігор існує досить широка кількість жанрів, вони діляться за візуальним виконанням, управлінням персонажа, цілями всередині гри, способом взаємодії з гравцем.

Наприклад, одним із досить популярних жанрів є платформер, де головним завданням гравця є успішне подолання внутрішньоігрового світу до мети, шляхом управління персонажем та використання його навичок пересування на кшталт стрибка, минаючи складнощі у обличчі ландшафту та ворогів. Ігри про найпопулярнішого відеоігрового персонажа у світі – Маріо – якраз і є платформерами. Шутер - ще один популярний жанр, де гравцеві потрібно знищувати цілі, наводячи на них найчастіше вогнепальну зброю та роблячи постріли. Прикладом такого жанру можна назвати успішну серію розрахованих на змагання між багатьма гравцями серію відеоігор Counter Strike.

Проект, що розробляється мною, відноситься до жанру екшн-рогалик. Це трохи нішевий, але й досить популярний жанр відеоігор. "Рогалик" походить від англійського "rogue-like" і означає "як Rouge", де Rouge - це гра 1980, де гравець досліджує підземелля. Відмінною рисою цієї гри стала випадкова генерація кімнат підземелля. Це означає, що кожного разу на початку нової гри підземелля, в яке потрапляє гравець, створюється з нуля за описаними грою правилами, таким чином гравець щоразу потрапляє до нового підземелля.

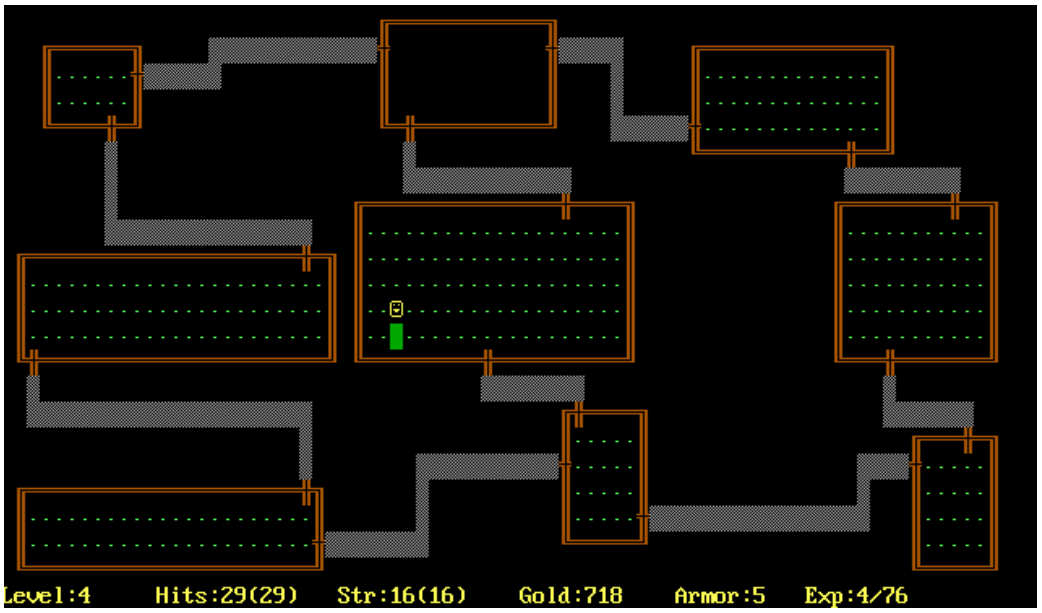


Рисунок 2.1 - Демонстрація відеогри Rouge

Суть ігор жанру рогалик полягає у реіграбельності. Іншими словами, коли більшість однокористувальних відеоігор дарують єдиний цілісний досвід проходження і закінчуються, коли гравець пройшов усі рівні, переміг усіх ворогів, зібрав усі предмети і так далі, рогалики навпаки припускають, що користувач гратиме в гру щоразу. Як правило, у рогаликах все-таки є кінцеві точки гри, проте якщо гравець все-таки до них доходить, то йому пропонується почати гру заново, як би по-колу, тому нова спроба гравця називається "забігом". При цьому якщо гравець не доходить до кінця гри за один забіг, він втрачає весь прогрес, який набрав усередині гри, і починає повністю з початку.

Однак тут велику роль відіграють ще стимули людини, що грає. Якщо раніше самого факту неповторності кімнат при кожній новій грі могло вистачити, щоб утримувати увагу користувача та змушувати його хотіти повертатися в гру ще й ще, то сучасного гравця такою ігровою механікою (правилом роботи інтерактивної взаємодії з грою) не здивувати. Для цього при створенні ігор у цьому жанрі користуються різними методами, що підвищують реіграбельність.

Наприклад, у таких іграх дуже велика складність. Складність – це якість гри, яка показує, наскільки у цю гру важко грати середньостатистичній людині. Чим вища складність - тим більше труднощів гра ставить перед гравцем, тим більше його розумових здібностей потрібно задіяти, тим сильніше вона вимоглива до його реакції та моторики, і тим сильніше гра карає користувача за помилки.

В іграх жанру рогалик потрібна висока складність для того, щоб розтягнути час між моментами, коли гравець лише входить у гру і коли гравець її проходить. Складність повинна бути такою, щоб гравець витратив десятки спроб, перш ніж пройти гру, поступово нарощуючи свої здібності. Досягнення кінця гри - не тільки мета для гравця, яку перед ним ставить відеогра, а й внутрішній стимул: подивитися, що буде в кінці, довести собі, що він може це зробити, що він здатний подолати завдання, які ставить перед ним гра. І такі психологічні стимули не варто недооцінювати, більше того, вони лежать в основі причин, чому люди грають у відеоігри.

При цьому складність у грі повинна бути чесною. Під цим зазвичай мають на увазі, що в переважній більшості випадків закінчення гри або смерті персонажа гравця всередині неї повинен бути винен сам гравець через свої дії. Це не означає, що гравця не можуть убити монстри, або перед ним не можна ставити важкі перешкоди, але у користувача завжди має бути можливість із цим впоратися. Правила, встановлені грою, повинні працювати завжди і не порушуватися, а якщо таке і відбуватиметься, то гра повинна про це повідомляти користувача у будь-який спосіб.

Розглянемо приклад: гравця атакує монстр та знімає йому дві одиниці здоров'я. Якщо в подальших випадках за таких самих умов монстр знову знімає дві одиниці здоров'я, то гравець запам'ятовує це як правило, що конкретний монстр віднімає у його персонажа конкретно стільки здоров’я, і цю установку гра не може порушувати. Розглянемо випадок, що у наступних випадках монстр знімає то один, то три, то знову дві одиниці здоров'я. У такому разі гравець сприймає правило так, що цей монстр знімає його персонажу від однієї до трьох одиниць здоров’я. Гравець приймає цю установку. Але якщо раптом чомусь без будь-яких на те причин рівно за таких же умов цей же монстр вб'є гравця з одного удару (тобто зніме, наприклад, 10 одиниць здоров'я), то гравець розсердиться.

І вина за це лежатиме не на ньому, а на грі, яка раптово вирішила його обдурити. В іграх з високою складністю у всіх смертях гравця або програші повинні бути винні тільки його власні неправильні рішення і дії, інакше гравець не бачитиме сенсу продовжувати грати в гру і ставати в ній краще, тому що його невдачі залежать не від нього, а від випадковостей, на які він не впливає. Не буде відчуття контролю та передбачуваності, заради якого багато хто і грає в ігри, у людини не буде стимулу вдосконалювати свої навички всередині гри, досягати поставлених цілей та розвивати свого персонажа, тому що у гравця буде думка про те що його можуть обдурити повторно і його перемога вже не залежатиме саме від нього.

За що дуже люблять рогалики, і за що в тому числі дуже зростає реіграбельність у таких відеоіграх, це за модифікатори персонажа. Ігровий

персонаж щоразу знову починає свій шлях у грі, але крім випадкових кімнат йому ще можуть траплятися випадкові речі, зброя, здібності і т.п., які якимось чином змінюють його характеристики. Наприклад, персонаж гравця може отримати додаткове здоров'я, швидше зброю або нову здатність. Як правило, таких модифікаторів у рогалику може бути дуже багато, і їх набір та послідовність отримування у кожному новому забігу відрізнятимуться один від одного, що стимулює гравця починати нові забіги для того, щоб використовувати нові комбінації модифікаторів, щоб подивитися, як його стиль та ефективність гри зміниться з новим набором предметів та як далеко він зайде.

Більш того, не всі модифікатори мають бути обов'язково очевидно вигідними для гравця.

Розберемо такий приклад: модифікатор, який знайшов гравець, збільшує кількість ворогів, але ще й додає шанс отримати додаткові аптечки з них. Тобто потенційно, цей модифікатор дозволяє гравцеві отримувати більше здоров'я і в перспективі зменшити шанс на загибель свого персонажа постійно його лікуючи, але водночас збільшується кількості ворогів – які є однією з причин можливої смерті персонажа – тобто можливість смерті збільшується, що ставить перед гравцем дилему: “Так, я отримуватиму більше аптечок, але чи зможу я впоратися з великою кількістю монстрів?”. Такі дилеми додають сильне різноманіття до ігрового процесу та підвищують залучення гравця до гри.

Жанр рогалик визначає те, як проходитиме ігровий процес і за якими правилами, дає зрозуміти про реіграбельність проекту і внутрішню варіативність, але все ще не дає відповіді на те, чим гравець безпосередньо в грі займатиметься. Тому жанр моєї гри - це екшн-рогалик, де слово екшн є англійським “action” (у перекладі “дія”). Екшн - досить широке поняття, і ігри цього жанру можуть дуже один від одного відрізнятися, однак їх поєднує сильний упор в динамічність ігрового процесу, його вимогливість до реакції, швидких дій гравця та його фізичних умінь.

Рисунок 2.2 - Демонстрація екшн гри Astral Chain

У деяких жанрах гравець може керувати кількома персонажами, цілими арміями, державами або знеособленими системами (на зразок міст або парків в іграх жанру тайкун). Екшн відеоігри більш звичні в цьому сенсі, тому що гравець у них керує найчастіше лише одним персонажем. Геймплей – спосіб, яким гравець взаємодіє з грою та впливає на неї – у таких іграх зосереджується на діях вашого персонажа: як він пересувається, як атакує, як захищається, що він ще вміє і як ці його особисті навички допомагають вам у проходженні гри.

В іграх жанру екшн-рогалик гравцю під управління потрапляє персонаж, який пересувається по віртуальному світу, бореться з ворогами ближньою або стрілецькою зброєю, проходить випадково згенеровані кімнати з випадковим набором ворогів і знаходить випадкові предмети та навички, комбінація яких повинна допомогти користувачеві пройти ігри, а при смерті весь прогрес обнуляється і доводиться грати вже у по-новому згенерованим рівням.

1.3 Аналіз існуючих популярних ігор в жанрі екшн рогалик

1.3.1 Nuclear Throne

Nuclear Throne - це однокористувацька гра в жанрі топ-даун (камера в грі знаходиться зверху над персонажем, ворогами та оточенням) шутемап (від англійського "shoot'em up", тобто "перестріляй їх усіх", піджанр екшену) рогалик, де гравцю потрібно керувати одним із мутантів і стріляти по різних ворогах, переміщуючись від однієї випадково згенерованої локації до іншої.

Гра спочатку була розроблена маленькою командою Vlambeer в 2014 році для одного з геймджемів - заходів для розробників ігор, де їм потрібно розробити маленьку гру в максимально стислий термін, в середньому протягом двох діб. Після хороших відгуків від журі та загальної зацікавленості аудіторії, творці доопрацювали прототип і випустили його в цифрових магазинах в 2015 році, і протягом трьох років оновлювали продукт, додаючи в нього новий контент і портуючи (процес переносу та адаптації гри з однієї платформи на іншу) версію для персональних комп'ютерів на платформи Xbox, Playstation та Nintendo. Гра коштує у цифрових магазинах дванадцять доларів США.

У грі персонаж може знаходити велику різноманітність стрілкової зброї, для якої використовують різні види снарядів, які теж потрібно гравцеві збирати. Однак носити з собою одночасно можна тільки два види зброї, що змушує гравця думати над максимально вдалим поєднанням наприклад повільної та швидкої зброї, далекобійної та зброї яка наносить сильний удар по області, а також комбінувати зброю так, щоб у гравця не було зброї однакового типу снаряди, інакше гравець може залишитися зовсім без патронів. На цьому прикладі можна побачити, що у грі також є і неочевидний тактичний елемент. Вбиваючи ворогів, гравець збирає мутаген - місцевий ресурс - за який може відкривати нові можливості для більш ефективного просування. Nuclear Throne можна ще виділити тим, що при проходженні всіх рівнів та вбивстві головного боса, гравець не закінчує гру, а локації починають йти по колу, але вже з набагато більшою кількістю та

різноманітністю ворогів. Таким чином, один забіг технічно може продовжуватися нескінченно, поки гравця не вб'ють. А реіграбельність ще підвищується за рахунок можливості грати за чотирнадцять персонажів, кожен з яких володіє унікальними здібностями.

Не варто ще й забувати про аудіо-візуальний досвід користувача. Гра виконана в піксельній стилістиці, але це не заважає їй виглядати стильно та по-аркадному динамічною. Над грою працював гарний художник Пол Вір. У той же час за музику і саунд-дизайн, що дуже запам'ятовується, відповідав композитор Юкіо Калліо: у кожної локації є по дві атмосферні теми, а звуки для всіх персонажів, ворогів та окремих видів зброї унікальні.

Рисунок 3.1 Демонстрація відеогри Nuclear Throne

У найпопулярнішому цифровому магазині ігор для персонального комп'ютера - Steam - Nuclear Throne має одинадцять тисяч оцінок, 97% з яких позитивний, а сама гра продалася понад півмільйона копій.

1.3.2 Hades

Hades - гра, розроблена невеликою американською компанією Supergiant Games і випущена у 2019 році, також є екшн-рогаликом. Гравець управляє принцом давньогрецького підземного царства Аїда – Загреєм.

Крім вже звичного геймплея з видом зверху у випадкових рівнях, розробники сконцентрувались на наративі (оповіданні), в результаті чого у головного героя є безліч діалогів з персонажами давньогрецького Пантеону, а багато ігрових умовностей пояснили через внутрішньоігровий світ або вплели їх у нього. Наприклад, всі рівні щоразу змінюються через те, що Аїд - батько головного героя - зачарував своє підземне царство, щоб ні його мешканці, ні головний герой не могли його покинути. Загрей – бог, тому й померти по-справжньому він не може, від чого щоразу після загибелі виходить із річки Стікс у палаці свого батька.

Персонажі запам'ятовують свої минулі розмови і відносини між ними навіть розвиваються від зустрічі до зустрічі, а безліч скриптів (сценаріїв, які виконуються за певних умов), на кшталт випадку, коли деякі боги дізнаються про конкретну зброю, з якою в даний момент гравець проходить рівень, і щось про нього розповідають, надають грі життя, допомагаючи із залученням гравця до продовження гри.

Такі аспекти Hades впливають на мета-прогресію, тобто на прогресію, яка не втрачається після смерті персонажа, адже вона є сюжетно обґрунтованою, і жодні ваші дії до неї не забуваються. Частиною цієї мета-прогресією також є сам палац Аїда, де персонаж може купувати різні предмети інтер'єру та модифікації, які не губляться після закінчення забігу. Це також є цікавим доповненням звичайного процесу гри, тому що всі предмети для інтер'єру купуються за внутрішню валюту, яку персонаж знаходить під час забігу. Валюта так само не губиться після смерті, що ставить гравця перед вибором: обрати валюту щоб потім щось купити для палацу, або обрати модифікатор для цього забігу.

Мета-прогресія в Hades допомагає гравцеві відчувати прогрес навіть після невдалих забігів, мотивуючи його не лише внутрішнім почуттям, а ще й ігровими можливостями та нагородами.

Графіка у грі виконана в унікальному візуальному стилі характерному також і для інших ігор цієї студії, при цьому сама гра поєднує 2D та 3D елементи. Музика поєднує в собі як і акустичні інструменти, що асоціюються з античністю, як баглами, так і важкі гітарні рифи.

Рисунок 3.2 Демонстрація відеогри Hades

Hades доступна на всіх актуальних відеоігрових платформах, коштує двадцять п'ять доларів США і розійшлася більш ніж один мільйон копій.

1.3.3 Soul Knight

Soul Knight – рогалик шутемап, що вийшов на мобільні платформи в ранньому доступі у 2017 році. Гра дуже натхненна Enter the Gungeon та Nuclear Throne, згаданої вище. На відміну від більшості інших ігор у цьому жанрі, наводитися на персонажів у Soul Knight не потрібно, гра робить це за гравця,

фокусуючись на найближчому ворогові. Тим не менш, сама гра від цього легкою не стає, адже зосереджуючи гравця на пересуванні персонажа та уворотах. Вона сама й розвиває цей напрямок, додаючи в ігровий процес буллет-хелл елементи (жанр, в якому гравцеві потрібно ухилятися від величезної кількості ворожих снарядів, вишикованих у різні геометричні патерни).

Але загалом темп гри трохи уповільнений порівняно з прикладами вище - все-таки розробники мобільних ігор орієнтуються на більш казуальну аудиторію.

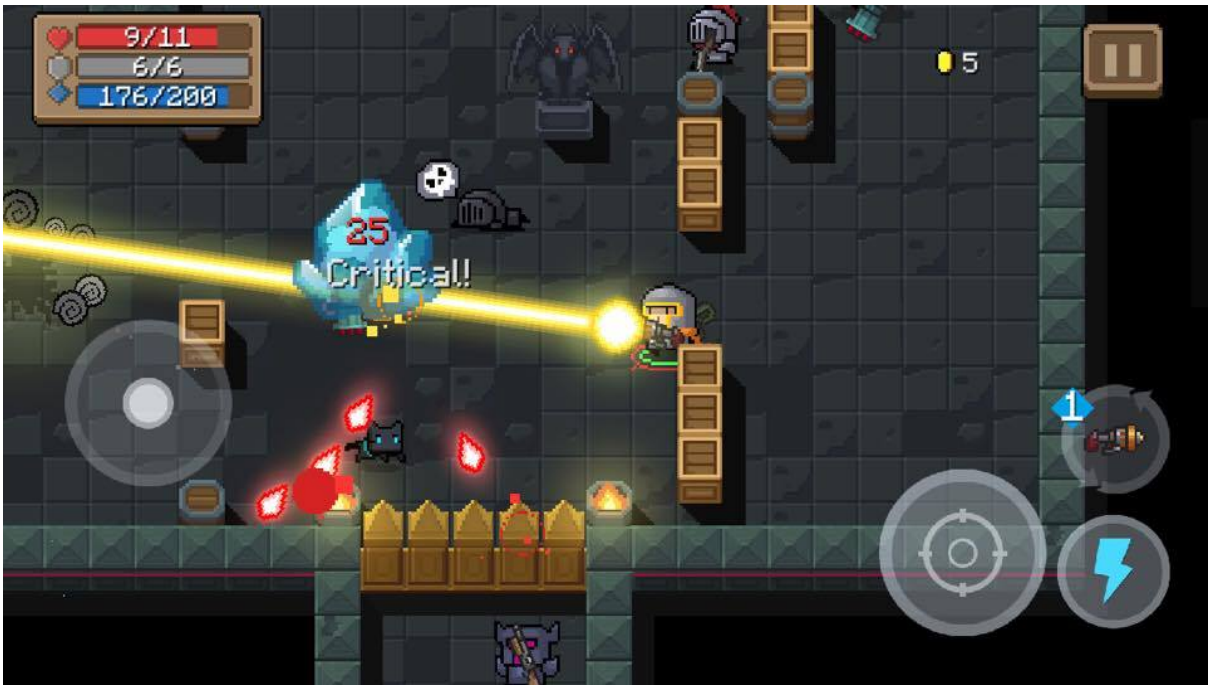


Рисунок 3.3 Демонстрація гри Soul Knight

Так само як і в Hades, в Soul Knight присутня мета-прогресія, яка втілена в можливості накопичувати внутрішньо ігрову валюту після кожного забігу і після купувати на неї нових персонажів, за яких можна грати, і розвивати сад з тваринами, який допомагає відпочити від забігів в основній частині гри.

У Soul Knight суттєві показники - понад п'ятдесят мільйонів завантажень тільки в Play Store. Гра розповсюджується за безкоштовною моделлю, але при цьому можливості заробітку для розробників все-таки присутні: наприклад, у грі можна відродити свого персонажа відразу після його смерті, переглянувши рекламний ролик усередині самої гри, і продовжити раніше провалений забіг. Як

вже було написано раніше, гравець дуже не любить втрачати прогрес навіть через власні помилки, а тому шанс того, що він вирішить подивитися рекламу заради єдиного шансу продовжити забіг зі своїми зібраними модифікаторами, дуже великий. Багато мобільних ігор заробляють за рахунок реклами у відведений час або надаючи якийсь функціонал всередині гри за її перегляд.

Більше того, у грі також присутні і внутрішньоігрові покупки: гравець може купувати персонажів, покращувати їх, купувати їм нові вигляди або якось прикрашати свій сад. Частину цього він може зробити за внутрішньоігрову валюту, яку можна накопичити просто граючи в гру, але для особливо нетерплячих і щедрих гравців завжди є можливість отримати все і відразу просто заплативши. Проте іншу частину контенту – близько половини ігрових персонажів та деякі їх вигляди – можна отримати лише за реальні гроші. Така система поширення гри називається "умовно безкоштовна".

Таблиця 3.1 - Порівняльні характеристики наведених ігор

Наведені ігри	Nuclear Throne	Hades	Soul Knight
Дата релізу	05.12.15	10.12.19	17.02.17
Платформи	Nintendo Switch, PlayStation 4, Xbox One, PlayStation 3, Windows, Linux, macOS, Mac OS, PlayStation Vita	Nintendo Switch, PlayStation 4, PlayStation 5, Xbox One, macOS, Xbox Series X|S, Windows, Mac OS	Android, iOS, Nintendo Switch
Наявність мобільної версії	Ні	Ні	Так
Вартість копії	12$	25$	Умовно безкоштовна
Внутрішньоігрові покупки	Ні	Ні	Так

Продажі (копій)	> 500 000	> 1 000 000	> 50 000 000
Складність геймплею	Висока	Висока	Середня
Наявність полегшеного режиму	Ні	Так	Ні
Мета-прогресія	Ні	Так	Так
Наявність різних грабельних персонажів	Так	Ні	Так
Якість аудіо-візуальної складової	Висока	Висока	Середня
Реіграбельність	Висока	Середня	Висока
Додатково	Комплексна онлайн таблиця рекордів	Хороший сценарій, працюючий наратив	Є локальний мультиплеєр
Ігровий двигун	GameMaker Studio 2	Внутрішній ігровий двигун студії Supergiant Games	Unity

1.4 Актуальність проекту та аналіз попиту

Екшн-рогаліки в даний час є досить популярним сегментом серед інди ігор, щороку виходять десятки ігор цього жанру, жодна сучасна відеоігрова виставка зараз не обходиться без рогаликів, а великі відеоігрові компанії теж намагаються створити свої проекти у схожому стилі (наприклад, Returnal видавництва Sony Playstation), що не може не казати про популярність таких проектів. Існує багато спільнот та інтернет блогерів, які присвячують свої матеріали виключно іграм цього жанру. І при всьому цьому популярність жанру не перестає зростати.

Жанр рогалика також користується великою популярністю на мобільних платформах, навіть у багатьох платних успішних проектах у магазині Play Store позначка досягає більш ніж сто тисяч завантажень, тоді як для багатьох безкоштовних ігор вона перевищує навіть п'ятсот тисяч. Звичайно, з гіперказуальними іграми на зразок ферм і три-в-ряд ці ігри навіть не варто порівнювати, але й аудиторія у екшн-рогаликів інша - зазвичай це діти та підлітки чоловічої статі віком від 8 до 18 років.

Досить великий відсоток серед цих ігор поширюються за умовно безкоштовною системою, коли сама гра безкоштовна, але її окремі елементи чи функції вже продаються або безпосередньо за гроші, або через внутрішньоігрову валюту, яку неможливо отримати у грі у достатніх кількостях. Хоча також вистачає і повністю безкоштовних ігор, монетизація яких обмежується показом реклами користувачам під час використання програми.

1.5 Об’єкт, предмет та мета проекту

Об’єкт дослідження - ігровий процес в жанрі екшн-рогалик.

Предмет дослідження - програмний продукт жанрі екшн-рогалик для мобільної платформи Android.

Мета роботи - Покращення користувацького досвіду із внутрішньожанровим різномаїттям відеогр типу екшн-рогалик для мобільних платформ за рахунок розробки гри із фокусом на близький контакт із ворогами.

1.6 Висновки до розділу

У розділі було розглянуто, що відеоігри - це програмне забезпечення, створене головним чином розважальних цілей, головна особливість якого у інтерактивності, тобто у взаємодії з користувачем – гравцем. Відеоігри почали з'являтися в кінці п'ятдесятих і на початку шістдесятих років минулого століття, і поступово розвивалися разом з технологічним прогресом, який сприяв створенню все більш потужних і компактних обчислювальних систем, які в свою чергу могли відтворювати більш складні та багаті на геймплей, графіку та аудіо ігри.

Ігри є розвагою і видом мистецтва, навколо якого існує величезна індустрія, і хоча по суті залишаються програмним забезпеченням, вони нерідко набагато ближче до фільмів, музики, книг. Як і інші медіапродукти, відеоігри можна поділити на категорії – жанри. Одним з таких жанрів є досить популярний у відносно вузьких колах екшн-рогалик, суть якого лежить у багатьох випадкових елементах і постійної повторюваності ігрового процесу. Були розглянуті одні з найпопулярніших проектів цього жанру.

Існує багато платформ для запуску на них відеоігор, і в наш час однією з найпопулярніших і найприбутковіших платформ є смартфони, які є буквально у кожної людини. Розробка проекту на мобільну платформу – досить популярний та прибутковий вибір для творців відеоігор.

Виходячи з усіх вищеописаних даних, я визначив і сформулював мету свого проекту – це створення мобільної гри жанру екшн-рогалик на ігровому движку.

2 АНАЛІЗ ЗАСОБІВ РОЗРОБКИ

2.1 Загальна характеристика ігрових двигунів

Для створення програмного забезпечення розробники завжди звертаються до будь-яких інструментів, наприклад, IDE (Integrated development environment), тобто середовище розробки. Для ігор найчастіше використовують ігрові движки - це програмне забезпечення, яке включає безліч готових бібліотек, рішень, інструментів, функцій, аддонів, підтримки різних ігрових платформ і зручного візуального представлення, де все з перерахованого створено спеціально для розробки відеоігор і спрощення цього процесу .

Наприклад, однією з основних функцій, які беруть на себе ігрові движки, є рендер зображення - процес надання візуальної частини гри, що відображається на дисплей пристроїв в реальному часі. Далі, у багатьох іграх об'єкти рухаються і взаємодіють за спрощеними фізичними законами, тому багато ігрових движків вже мають мінімальний вбудований фізичний движок. Якщо людина або команда буде створювати свою гру без ігрового двигуна, їм знадобиться розробляти та прописувати всі ці та інші речі повністю вручну.

Деякі ігрові студії розробляють ігрові двигуни для власного внутрішнього користування. Такими, наприклад, є X-Ray Engine української студії GSC Game World, на якому вийшло три ігри, REDengine від польських CD Projekt RED з чотирма іграми, RAGE, розроблений студіями Rockstar San Diego та Rockstar North, з тринадцятьма і так далі.

При цьому існує безліч незалежних ігрових двигунів у відкритому доступі, кожен з яких має плюси та мінуси. Існують як і движки, призначених для розробки широкого спектру проектів, так і вузькоспеціалізовані, чий функціонал зосереджений на створенні додатків одного жанру, для однієї платформи, одного виду графіки і т.д.

Найголовнішими і найпопулярнішими ігровими двигунами зараз є Unreal Engine і Unity, але це не означає, що всі ігри створюються тільки тільки на них.

Також досить популярними вважаються Godot, GameMaker Studio, CryEngine та багато інших. Як варіанти для розгляду були взяті Unreal Engine, GameMaker Studio 2 та Unity.

2.2 Аналіз ігрових двигунів

2.2.1 Unreal Engine 4

Unreal Engine 4 – один з найпопулярніших ігрових движків, розроблений ігровою компанією Epic Games. Перша версія інструменту була представлена в 1998 році, а найновіша - Unreal Engine 5 - вийшла на початку 2022 року. Тим не менш, найбільш ходовий і все ще актуальною є четверта версія двигуна, випущена 2014 роком.

Спочатку двигун призначався для створення шутерів від першої особи, проте змінюючись і обростаючи функціоналом з кожною наступною версією, став застосовуватися для розробки найрізноманітніших проектів.

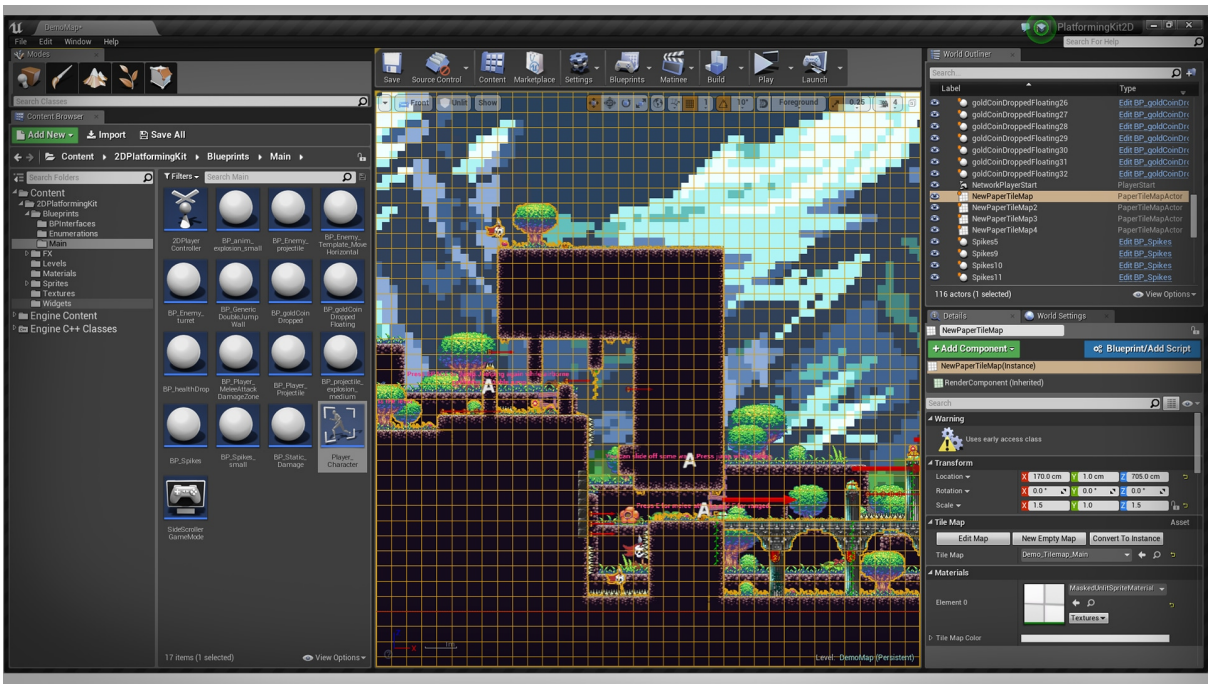


Рисунок 2.1 - Демонстрація двигуна Unreal Engine

Раніше Unreal Engine поширювався по передплатній системі, проте з 2015 року перейшов на більш хитру систему, при якій отримання та використання двигуна повністю безкоштовні, але розробник зобов'язується платити відрахування з виручки за проекти, розроблені на Unreal Engine.

Сам ігровий движок написаний мовою програмування C++, тому файли для додавання власної ігрової логіки теж пишуться цією мовою. Для більш спрощеного підходу у створенні в движку існує система блюпринтів. Блюпринти (англійське blueprints) - це скриптова система, що є візуальним інтерфейсом ігрової логіки, де все будується з блоків з об'єктами та їх характеристиками та зв'язками між ними. Такий спосіб розробки створено для швидкої розробки елементарних речей або просто людей, які не вміють програмувати, але хочуть створювати ігри. Тим не менш, досвідчені розробники все ж таки воліють користуватися C++, так як за допомогою нього можна змінювати базовий двигун і робити набагато більш гнучку логіку з оптимізацією, адже мова відносно низькорівнева, а відеоігри - досить вимогливе до параметрів заліза програмне забезпечення.

І все ж таки двигун більше розрахований на роботу з тривимірною високополігональної графікою, з безліччю ефектів і постобробкою. На Unreal Engine є двовимірні проекти, але двигун із самого початку свого існування створювався для розробки 3D ігри

Плюси:

- Безкоштовний движок
- Багатий функціонал
- Постійна підтримка та безліч додатків для користувача
- Інструменти для роботи зі штучним інтелектом
- Безкоштовні асети у великих кількостях

Мінуси:

- Не застосовується для створення 2D ігор
- Двигун погано підходить для невеликих за обсягом проектів
- Задіює багато обчислювальних ресурсів в грі

2.2.2 GameMaker Studio 2

GameMaker Studio 2 - дуже популярний ігровий двигун, перша ітерація якого вийшла в 1999 році, а поточна була випущена в 2012 році і ще довго оновлювалася.

GameMaker заплутана і складна модель поширення: на сайті движка є безкоштовна версія для ознайомлення, в ній можна навіть зробити ігри для їх сайту, але на цьому функціонал обмежений. Є кілька рівнів щомісячних підписок з різним рівнем функціоналу та можливості портувати на різні платформи. Вартість передплат варіюється від п'яти до вісімдесяти доларів США. При цьому в цифрових магазинах є повноцінні окремі версії ігрового двигуна для розробки тільки на десктоп, тільки під мобільні платформи і лише під веб-розробку. Коштують такі версії за сто доларів США.

Головним чином для розробки на GameMaker використовується спрощена мова програмування GML (Game Maker Language) власної розробки, що також бере участь і у “візуальному” режимі - аналогі блюпринтів у Unreal Engine. Вбудований модуль YoYoCompiler транслює код, написаний на GML, C++ і навіть додатково його оптимізує, що дозволяє досягти високої продуктивності ігор.

На відміну від попереднього ігрового двигуна, GameMaker Studio 2 підтримує створення лише двовимірних ігор, і весь його функціонал спрямований саме на це.

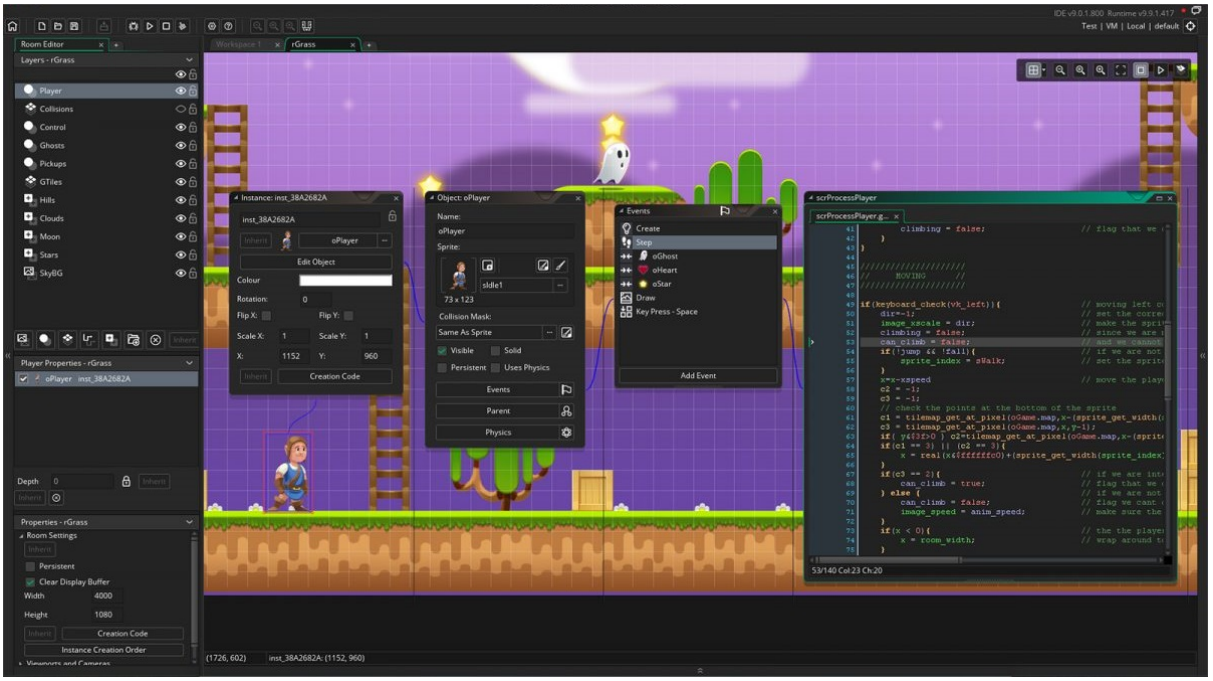


Рисунок 2.2 - Демонстрація двигуна GameMaker Studio 2

Серед успішних рогаликів, розроблених на GameMaker Studio 2 можна виділити розібраний раніше Nuclear Throne, а також Space Robinson і Spelunky.

Плюси:

- Гнучка цінова політика
- Мультиплатформеність
- Спрямованість на розробку невеликих ігрових проектів,
- Легка мова
- Докладна документація

Мінуси:

- Вартість
- Відсутність підтримки програмного продукту
- Скромна кількість доступних встановлюваних аддонів
- Менший доступний функціонал порівняно з конкурентами.

2.2.3 Unity

Unity - ігровий двигун, розроблений компанією Unity Technologies і вперше випущений у 2005 році. Як і інші двигуни, Unity підтримує мультиплатформні проекти, має широкий функціонал разом із зручним графічним інтерфейсом.

У порівнянні з раніше наведеними двигунами, Unity використовується для створення 2D ігор так само часто, як і для 3D, причому інструмент популярний як для невеликих проектів, так і для масштабних. У програмі повно інструментів для роботи з двома видами вимірювань, є вбудований фізичний двигун, наявність функціоналу штучного інтелекту для негірабельних персонажів та багато інших корисних функцій.

Unity має безкоштовну версію, але якщо користувач випустить успішний комерційний продукт, зроблений за допомогою движка, то йому потрібно буде оформити річну ліцензію, ціна якої залежить від прибутку з проекту. Інакше двигуном можна користуватися абсолютно безкоштовно.

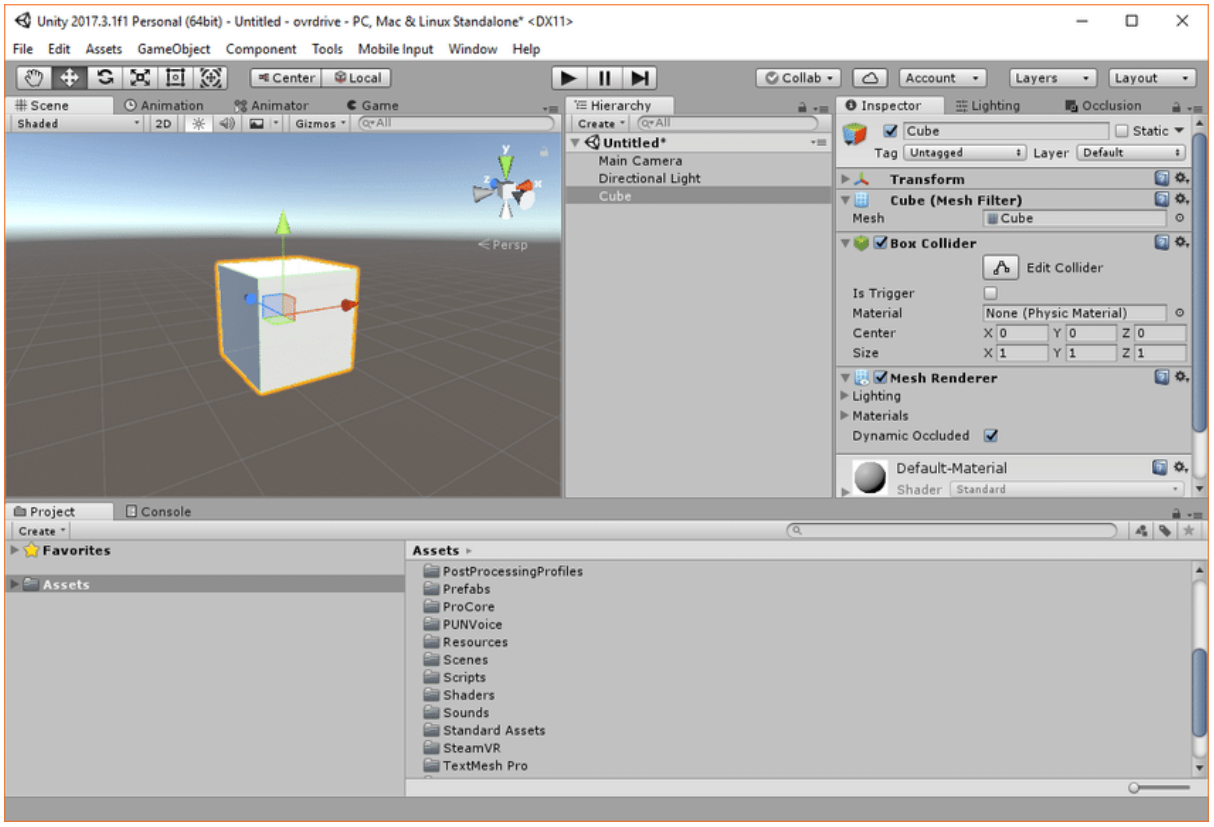


Рисунок 2.3 - Демонстрація двигуна Unity

Серед популярних рогаликів, створених за допомогою Unity, можна назвати Enter the Gungeon, Risk of Rain 2, Soul Knight.

Плюси:

- Мультиплатформенність для проектів
- Велика кількість інструментів та аддонів
- Каталог асетів
- Підтримка як 2D так і 3D
- Безкоштовний для розробки
- Інтеграція з іншими програмами для розробки.

Мінуси:

- Погана оптимізація на великих сценах

2.3 Середа розробки програмного забезпечення

Крім самого ігрового двигуна також знадобиться IDE для написання коду програми, адже сам двигун такої можливості не надає. Є варіант використання звичайних редакторів коду, на зразок Visual Studio Code, або навіть звичайних текстових редакторів, але функціонал і представлення даних та зв'язків у середовищі програмування беззастережно є великим плюсом для комфортної розробки.

Так як найбільш оптимальним ігровим движком для створення задуманої гри був визначений ігровий движок Unity, який використовує мову програмування, то і IDE має бути відповідним. Тому було обрано найпоширеніша IDE для написання програм мовою C# - Microsoft Visual Studio.

Більш того, існує інструмент інтеграції архітектури двигуна в середу Visual Studio, після чого при написанні коду VS розпізнає бібліотеки Unity.

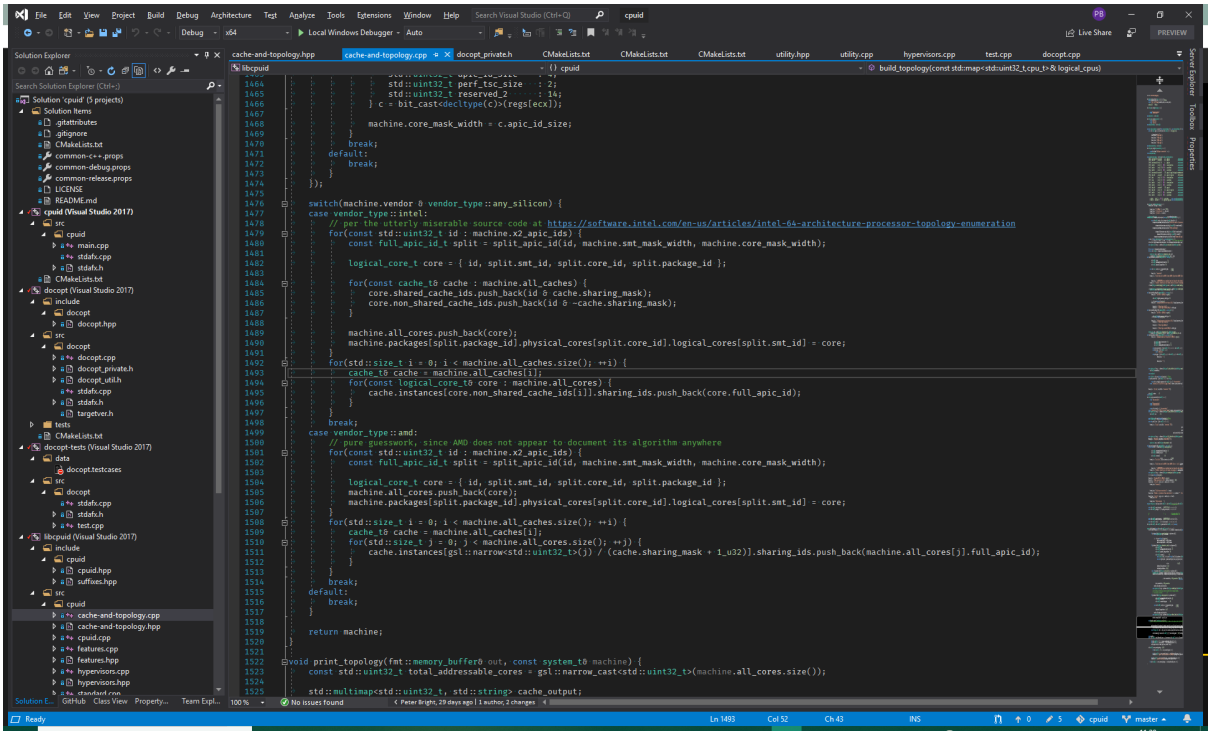


Рисунок 2.4 - Демонстрація IDE Microsoft Visual Studio

2.4 Графічний редактор

Для створення гри також знадобляться спрайти - графічні об'єкти, які відображатимуться на екрані під час гри та символізуватимуть ті чи інші ігрові об'єкти.

Як було написано раніше, Unity має зовнішній каталог ассетів, у тому числі для графіки. Окрім самого каталогу, знайти ассети з графікою можна і у вільному доступі в інтернеті, як безкоштовно, так і за певну ціну, для цього навіть створюються спеціалізовані цифрові магазини та сайти.

Тим не менш, маючи досвід у створенні комп'ютерної графіки, у тому числі для різних ігрових проектів, я вирішив створити усі необхідні графічні матеріали самостійно. Візуальний стиль гри буде виконаний у стилі піксель-арт (це коли зображення навмисно пікселізовано для стилізації або отримання ефекту відчуття ретро гри). Тому серед широкого спектру графічних редакторів було обрано Aseprite – програмне забезпечення 2001 року, створене та заточене спеціально для роботи з піксельною графікою.

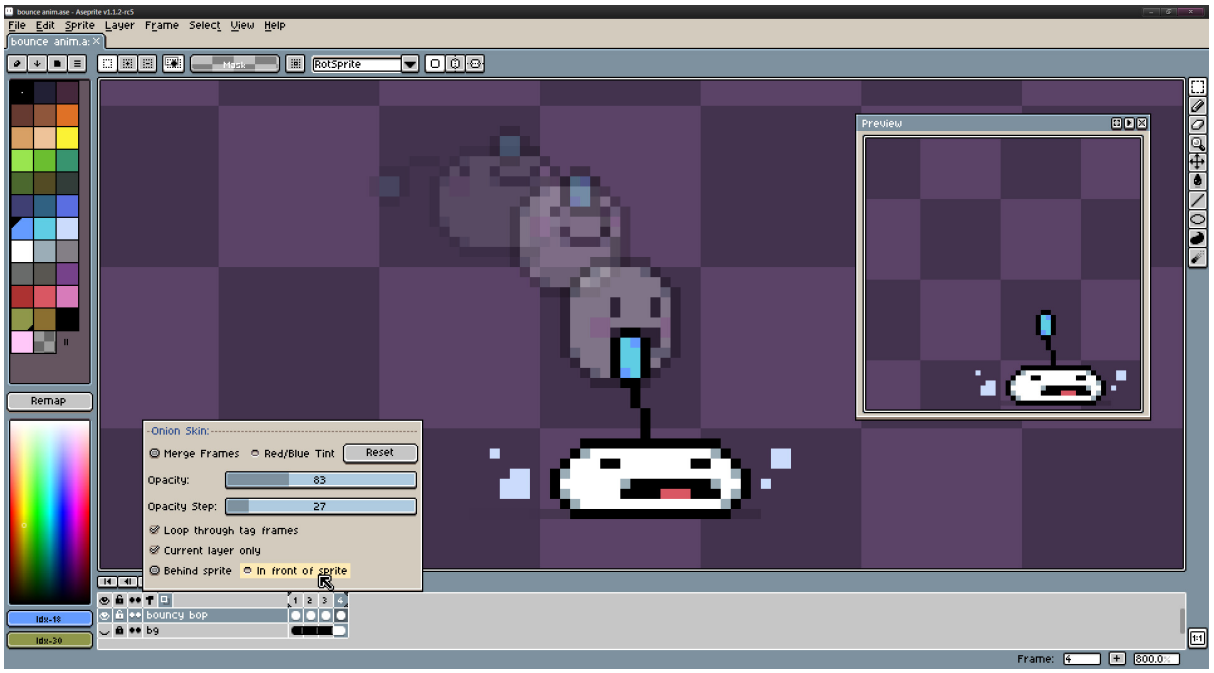


Рисунок 2.5 - Демонстрація графічного редактора Aseprite

Програма дозволяє створювати та редагувати піксельні зображення та анімації, експортувати зображення та працювати з безліччю розширень. Також підтримує адони для нового функціоналу.

2.5 Висновки до розділу

У цьому розділі було розглянуто засоби розробки, які знадобляться для реалізації гри. Проаналізувавши ігрові двигуни, було визнаено, що для розробки 2D проекту найкраще підходять Unity і GameMaker Studio 2, але у першого більша кількість можливостей та готових рішень, спрощена система портування ігор на різні платформи. Для розробки на движку Unity використовується мова програмування C#, тому безпосередньо для написання коду була обрана відповідна IDE - Microsoft Visual Studio, яка також має інтеграцію з обраним ігровим движком Unity. Піксельна графіка для проекту була обрана через легкість та швидкість її створення, популярність та наявність досвіду роботи з нею у мене. Тому і графічний редактор був обраний спеціально для роботи з піксельною графікою – Aseprite.

3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ МОБІЛЬНОЇ ГРИ

3.1 Планування гри та дизайн документ

Отже, перш ніж розробляти будь-який програмний продукт, слід визначитися, що буде в цьому продукті. Іншими словами, необхідно підготувати дизайн документ - точний, детальний і структурований опис всієї гри.

Назва гри: Undergrounder

Жанр: Екшн рогалик

Платформа: Android

Цільова аудиторія: чоловіки віком 8-18 років

Модель поширення: Платна, 2,5 $

Короткий опис ігрового процесу: Гравець щоразу потрапляє до мережі по-новому згенерованих рівнів. На рівні персонажу потрібно знищити всіх ворогів, отримати поліпшення і перейти на наступний рівень. Кількість випадково згенерованих рівнів необмежена, у кожному наступному рівні складність підвищується за рахунок кількості та типів ворогів. Весь прогрес губиться, коли персонажа вбивають.

Мета гри: Дійти до якомога більшого рівня

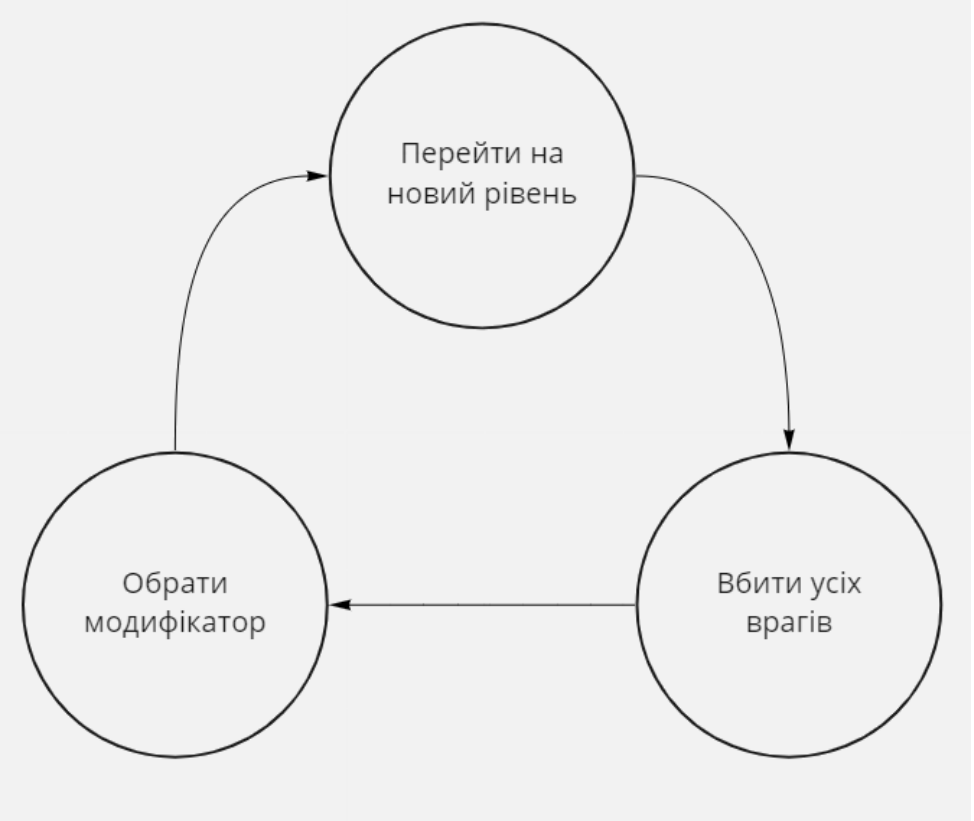


Рисунок 3.1 - Діаграма Core Loop гри

Візуальний стиль: Яскрава піксельна 2D графіка та низька кількість кадрів в анімаціях для легшого зчитування ворогів. Вид зверху.

Місце дії: Підземні станції метро.

Персонаж: Маленький кріт.

Здібності персонажа: Пересування по площині, удар зброєю, деш (миттєве переміщення персонажа на невелику дистанцію для ігнорування втрат), можливість міняти зброю, можливість вибирати модифікатори, піднімати аптечки

Вороги: Різні мутанти, що мешкають у підземному метро.

Опис базової поведінки ворогів: Вороги спочатку ходять на невеликі випадкові відстані з зупинками, при попаданні персонажа гравця в зону видимості ворога, останній переходить в активну стадію, переслідуючи і намагаючись завдати гравцю шкоди.

Модифікатори: Модифікатори будуть фруктами, і в кінці рівня персонажу потрібно буде вибрати, який саме фрукт він хоче отримати. Фрукти впливатимуть

на швидкість пересування, швидкість атаки, кількість монстрів, шанс випадання аптечки, зниження демеджу.

Рівні: Рівні генеруються випадковим чином за заданим алгоритмом, мають витягнуту прямокутну форму з перешкодами та ворогами. Кожен наступний рівень складніший за попередній рахунок підвищення шансу появи перешкод, більш важких ворогів та їх кількості.

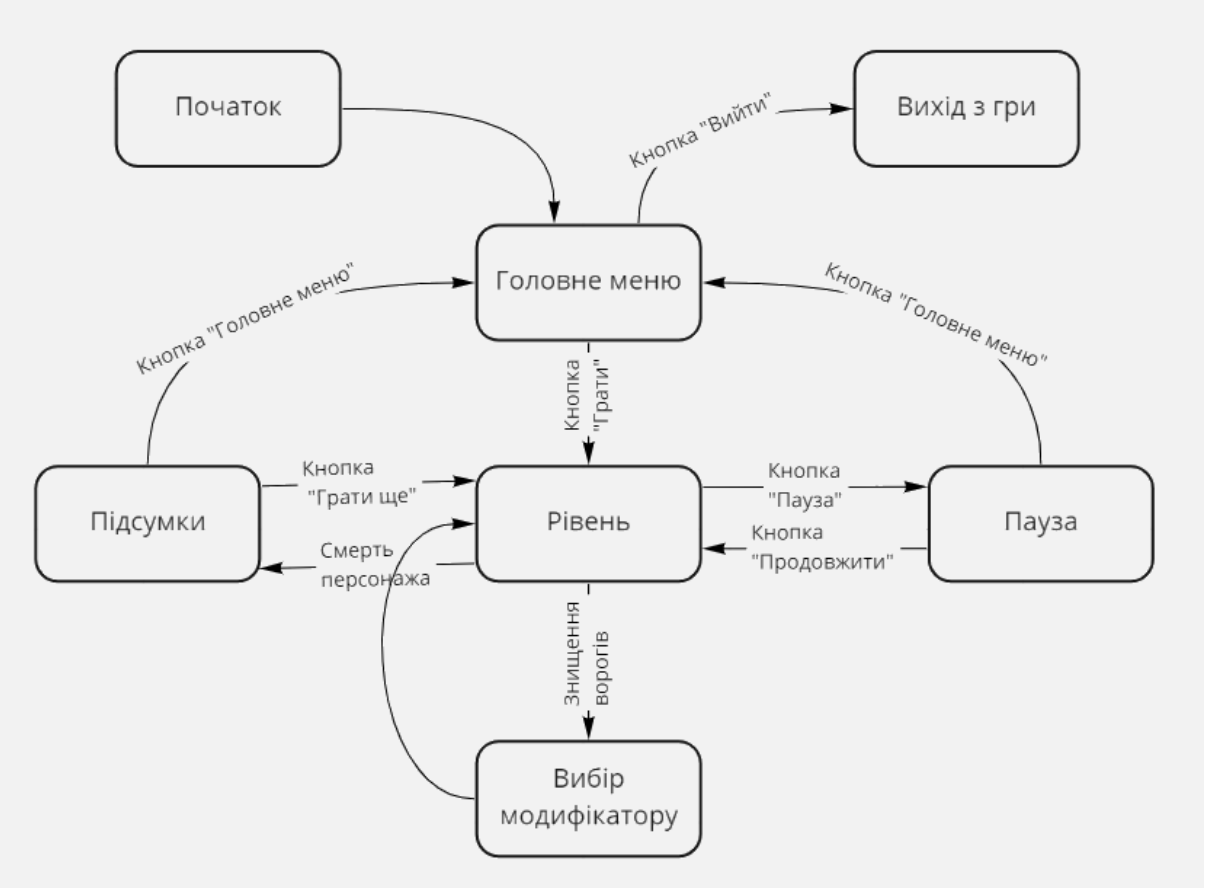


Рисунок 3.2 - Діаграма екранів гри

3.2 Створення прототипу

Коли я створив дизайн документ і описав всі основні характеристики та механіки гри, я можу приступити до створення прототипу - базової тестової версії гри, завданням якої є продемонструвати основні механіки ігри для того, щоб протестувати, як вони працюють на базовому рівні. Після створення прототипу, на

нього нарощуватиметься решта функціоналу гри з додаванням решти речей і полірування вже наявних.

У прототип входитиме:

- Персонаж зі здатністю ходьби
- Зброя, що завдає шкоди
- Ворог, що атакує персонажа
- Простий рівень зі стінами

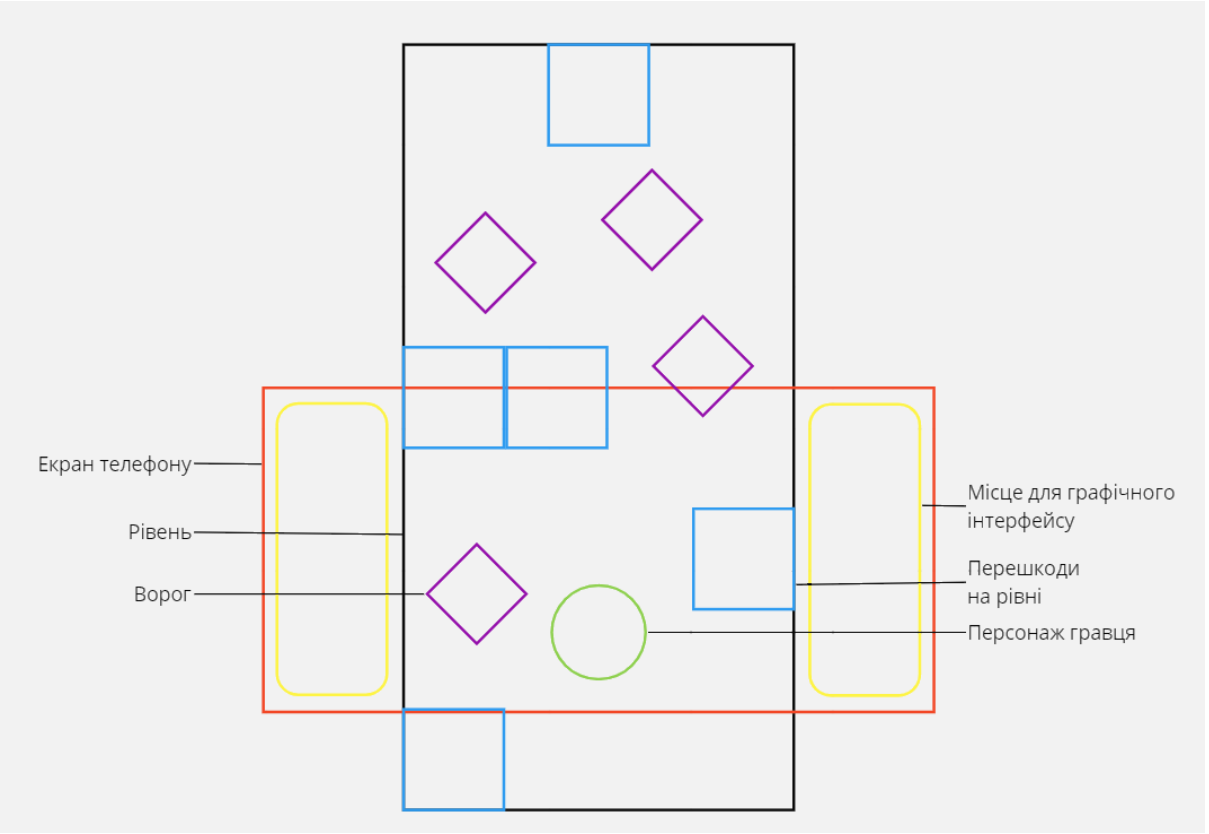


Рисунок 3.2 - Схема гри під час проходження рівнів

Насамперед, потрібно відкрити Unity, вибрати версію двигуна і створити новий Mobile 2D проект, після чого відкриється вікно розробки нового проекту, де вже можна приступати до роботи.

Для початку потрібно створити персонажа - базову геометричну фігуру, прописати їй здоров'я можливість ходьби та атаки.

Тому що я створюю прототип - я поки що можу не думати про дерево успадкування класів, і поки що створити тільки три класи:

1. Entity - на даному етапі є клас істоти, що відповідає за можливість персонажем рухатися, надалі буде сполучною ланкою всіх класів на істоті
2. PCInputController - клас, який відповідає за обробку натискання клавіш клавіатури, тому що для розробки прототипу поки що вистачить і комп'ютерної версії
3. HealthController - клас, який відповідає за здоров'я, шкоду та знищення об'єкта

Після створення об'єкта всередині гри на сцені нам потрібно додати йому компоненти - це Rigidbody 2D, який відповідає за фізичну взаємодію з об'єктом, масу, силу гравітації (яку через особливості проекції відеоігри я повинен поставити на 0, інакше персонаж відлетить вниз за екран) та інші параметри, головним чином які нам потрібні для налаштування пересування персонажа, тому що рухати буду його не за допомогою координат, а за допомогою привласнення його фізичному тілу сили в конкретному напрямку, що власне і рухатиме персонажа. Collider 2D - компонент, який створює і налаштовує межі взаємодії ігрового об'єкта, по них судитиметься, чи провзаємодіяв об'єкт з іншим, чи стосується його, чи потрапило в нього джерело збитків і так далі. При додаванні Sprite Render з'являється можливість встановити об'єкту графічне відображення для того, щоб цей об'єкт при грі бачили і він мав конкретну візуалізацію. Для початку можна використати просто кольоровий квадрат.

Після цього треба додати компоненти наших скриптів, описаних вище, і виставити їм потрібні параметри. У Unity є чудова можливість - при помітці поля атрибутом "SerializeField", значення цього поля можна виставити прямо у вікні движка.

Рисунок 3.3 - Робота з серіалізованими полями в Unity

Виконуються аналогічні дії з ворогом і зброєю, причому об'єкт зброї поміщається в об'єкт гравця, таким чином перебуваючи в його ієрархії і будучи дочірнім елементом. Це означатиме, що гравець - точка відліку зброї, і під час руху гравця зброя переміщатиметься разом із нею.

А ось для створення самих рівнів, надалі їх генерації, розстановки на них перешкод і стін буде використовуватися Grid (сітка) та TileMap – файл з набором можливих малюнків, у даному випадку для оточення та рівня. Для більш простої геометрії рівня і відповідно легшого алгоритму її створення, будується використовувати сітку з тайлами (плитками), таким чином рівень буде схожий на лист зошити в клітинку, де різні клітини будуть по-різному зафарбовані відповідно до свого призначення - будь то підлога , стіни, перешкоди або пастки.

Для взаємодії персонажа зі стіною, а точніше впирання персонажа в стіну, використовуються якраз коллайдери. Надалі їх можна буде гнучко налаштувати, включаючи таблицю фізичної взаємодії між різними шарами гри. Все це дасть можливість упиратися в деякі об'єкти, на кшталт стін та перешкод, а інші – проходити наскрізь, як тайли поверхні підлоги.

А ось вже для взаємодії двох проходять один-одного наскрізь об'єктів, можна вже використовувати вбудований метод OnTriggerEnter2D, який виконуватиметься щоразу, коли один коллайдер перетинає інший. Як аргумент OnTriggerEnter2D приймає колайдер об'єкта, з яким була взаємодія, і вже через нього можна достукатися до самого об'єкта і, наприклад, викликати у нього якийсь метод. Саме так ворог і отримає шкоду, коли його колайдер стикнеться з колайдером лопати. Аналогічну логіку прописую і для отримання шкоди персонажем від ворога.

Прототип готовий, тепер можна перейти до подальшої розробки.

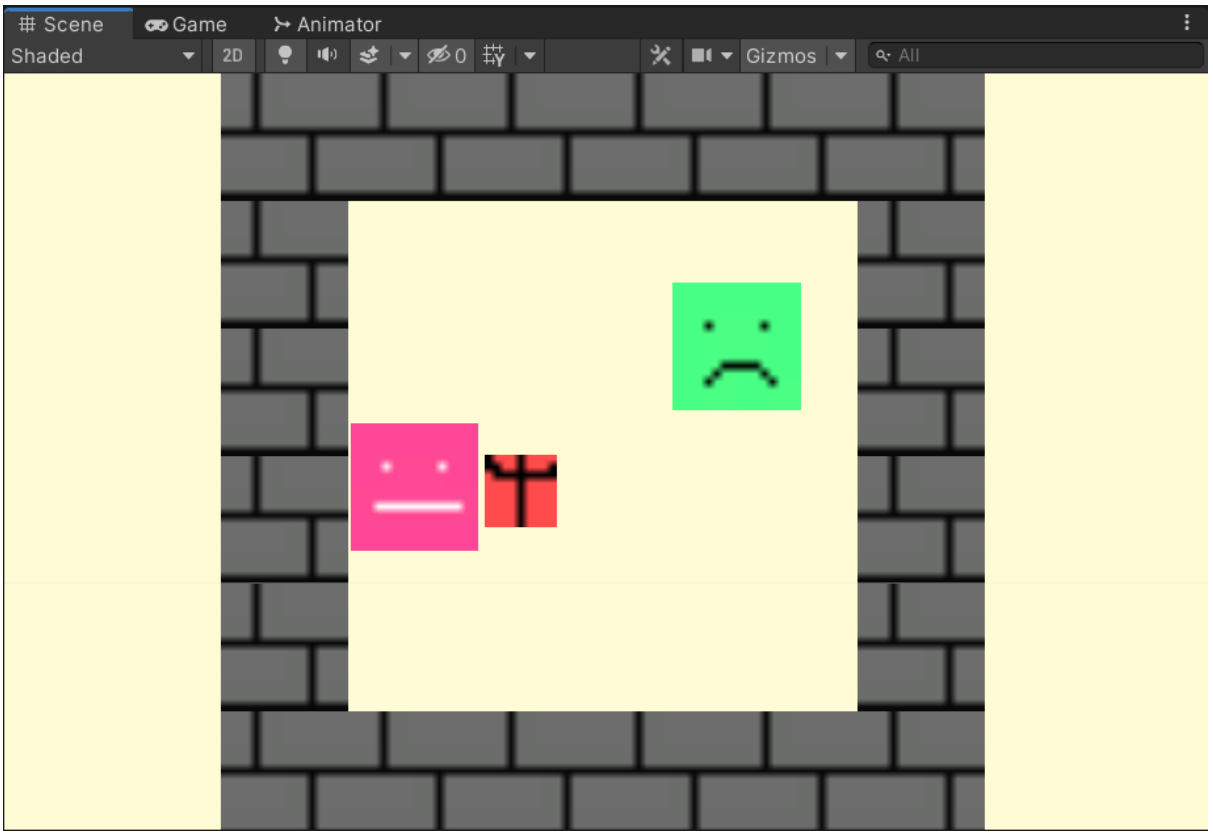


Рисунок 3.3 - Прототип гри

3.3 Створення графіки

Як було визначено в дизайн документі, візуальний стиль гри характеризуватиметься піксельною графікою маленької роздільної здатності з яскравими кольоровими елементами та простими анімаціями.

За основну міру створення графічних об'єктів братиму шістнадцять пікселів - саме стільки займатимуть клітини рівня. Тому й персонажа створюю у межах цих розмірів. Також додаю і анімації для деяких дій та станів персонажа – стан спокою, ходьба, отримання шкоди, лікування та смерть. Всі ігрові об'єкти будуть повернені до гравця обличчям, тому їх, разом з персонажем, можна намалювати лише з одного боку, а всередині гри розгортати їх ліворуч програмним шляхом, обертаючи спрайти істот навколо їхньої осі Y.

Рисунок 3.4 - Процес створення головного героя та його анімацій

Далі, відштовхуючись від нього, створюю тайли для оточення, ворогів та зброю. Кольори та форми ворогів у швидких динамічних іграх повинні бути легко зчитуємими та відрізнятися один від одного та об'єктів оточення, щоб гравцеві було відразу їх видно і він міг легше та швидше оцінити обстановку у грі та вжити потрібних дій. Фон, а точніше тайли підлоги, будуть не такими контрасними, як живі вороги, а також не буде використовуватися контур. Знов-таки, це все для досягнення читання персонажа та ворогів.

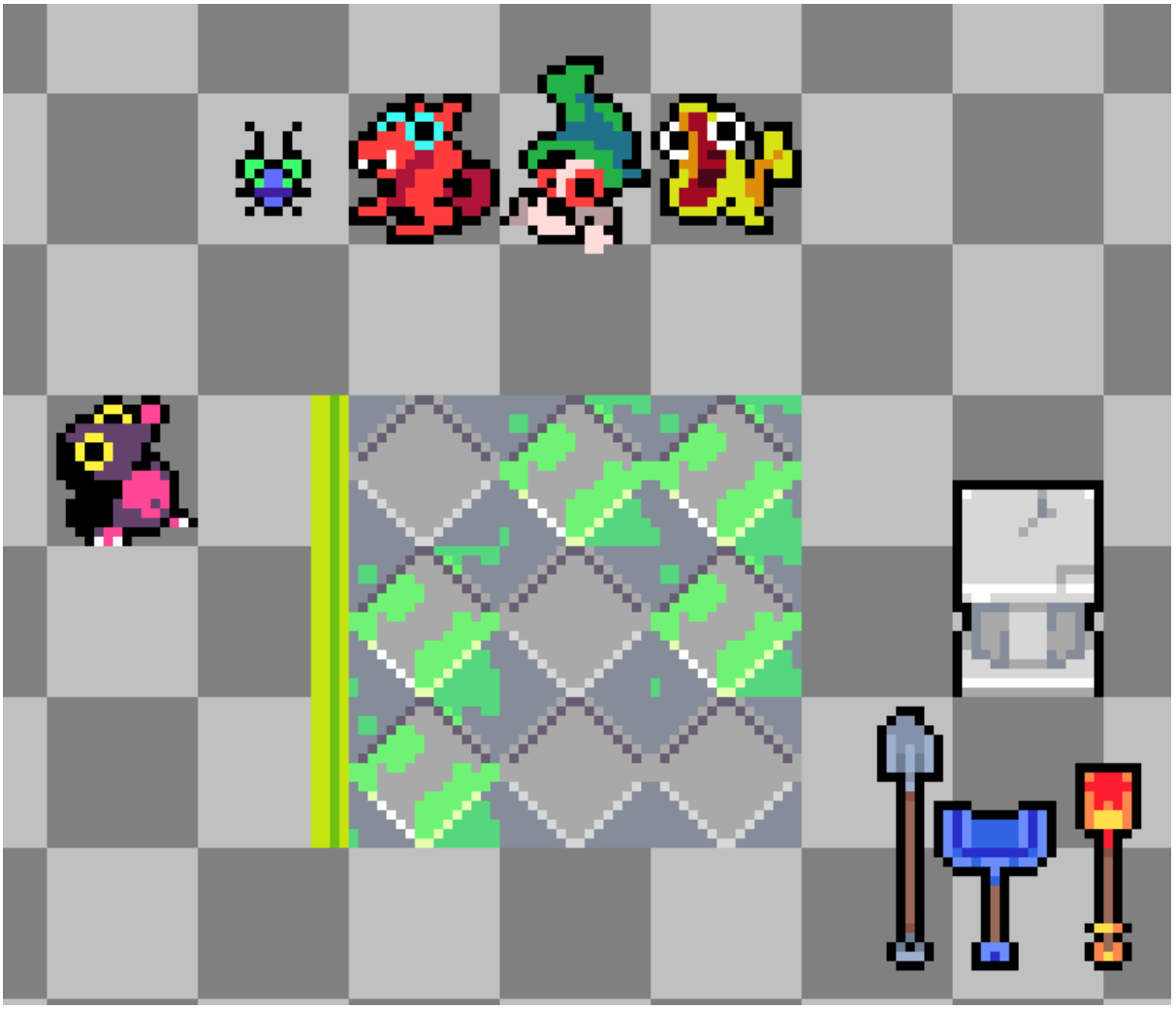


Рисунок 3.5 - Процес створення графіки для гри

3.4 Реалізація відеогри

Тепер, коли я вже маю достатньо матеріалів для додавання їх у гру, можна зайнятися подальшою розробкою та нарощуванням функціоналу. Насамперед варто створити головне меню. Для цього потрібно створити нову сцену. Як було зображено на діаграмі екранів, у меню мається дві кнопки: “Грати” та “Вийти”. Так як функції в різних ігрових меню часто повторюються і просто змінюють ігровий екран, сам процес зміни сцен буде здійснюватися в класі ServiceManager, який матиме патерн Singleton, який гарантує наявність тільки одного екземпляру класу, а ось решта меню матимуть до цього екземпляра доступ . Для роботи зі сценами також потрібно підключити бібліотеку UnityEngine.SceneManagement.

```
#region SingleTon
public static ServiceManager Instance;
Сообщение Unity | Ссылок: 0
private void Awake()
{
    if(Instance == null)
    {
        Instance = this;
    }
    else
    {
        Destroy(gameObject);
    }
}
#endregion
```

Рисунок 3.6 - патерн програмування “Singleton”

Для роботи будь-якого меню мені знадобиться власний абстрактний батьківський клас - BaseMenuController, в якому встановлюватиметься посилання на екземпляр ServiceManager, а також прописуватиметься базовий функціонал будь-якого меню. Наслідуючи клас BaseMenuController, я можу вже створювати приватні класи для конкретних екранних меню. Таким чином створюю MainMenuController, де я прописую поля спеціально для окремих кнопок, після чого через вікно двигуна поміщаю в поля вже створені на сцени елементи Button, а в самому скрипті підписую ці кнопки на потрібні методи класу BaseMenuController або ServiceManager. Так само будуть створені і всі наступні меню.

Тепер для створення нових рівнів я напишу скрипт, який буде створювати їх планування - LevelStructureGenerator. Оскільки рівень є прямокутною сіткою, можно створювати за певним алгоритмом двовимірний масив з чисел, де 1 відповідатиме за стіни по контуру двовимірного масиву, 2 буде підлогою а 3 перешкодою. Для цього я активно користуватимусь випадковими числами, але в певному діапазоні і з обмеженнями. Потрібно, наприклад перевіряти, не будуть

перешкоди стояти суцільною лінією і тим самим загороджувати частину рівня, що залишилася, а довжина рівня повинна бути в межах від 15 до 30 тайлів. Після цього також буде прораховуватися випадковий набір монстрів, на яких як ігрові об'єкти, а точніше їх пресети, будуть заготовлені посилання. Набір, а точніше кількість монстрів різних типів теж залежатиме від коефіцієнтів. Через великі обсяги логіки як у генерації рівня, так і підбору ворогів, створюю окремий клас для останнього - LevelEnemyGenerator. Цей скрипт також розставлятиме монстрів на рівні, перевіряючи наявність на координатах перепон і близькість до координат появи гравця. З кожним наступним рівнем шанс того, що рівень буде все довшим, а монстрів більшим, збільшується, це можна робити, додаючи до шансу коефіцієнт, який обчислюватиметься на основі порядкового номера рівня, що проходить.

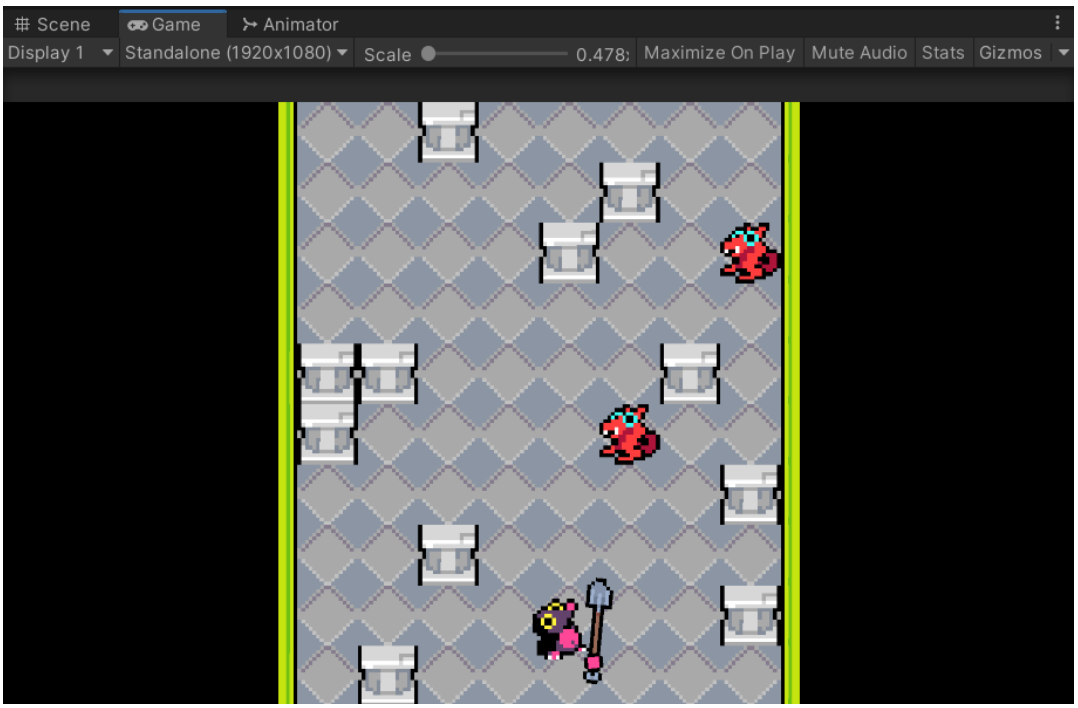


Рисунок 3.7 - Демонстрація сгенерованого рівня

Після того, як масив сформований і перевірений на можливі помилки, Скрипт розставляє тайли по координатах відповідно до сітки рівня, при чому одні тайли відносяться до одного шару і колайдера не мають, а інші вже будуть його

мати. Далі за такою ж логікою розставляються пресети ворогів та інше. Скрипт LevelManager відстежує наявність всіх ворогів на рівні, і коли нікого з них не залишається, викликає меню вибору модифікатора – фрукта. Після вибору фрукт екіпірується гравцю, а скрипт запускає перестворення рівня, де збільшується його порядковий номер і все відбувається по-новому.

Далі йде опрацювання істот. Потрібно створити окремі базові абстрактні класи BaseEntity, який буде контейнером всіх складових компонентів, і BaseEntityActionController, який відповідає за стани і дії істоти. Також необхідно створити тип, що перераховується, з видами дій, за якими буде перемикатися логіка дій істот. Наслідуючи ці класи, я створюю аналогічні для персонажа гравця та ворогів. Також потрібно додати клас, який зберігатиме та обробляти характеристики істот, на кшталт кількості здоров'я, швидкості, захисту тощо.

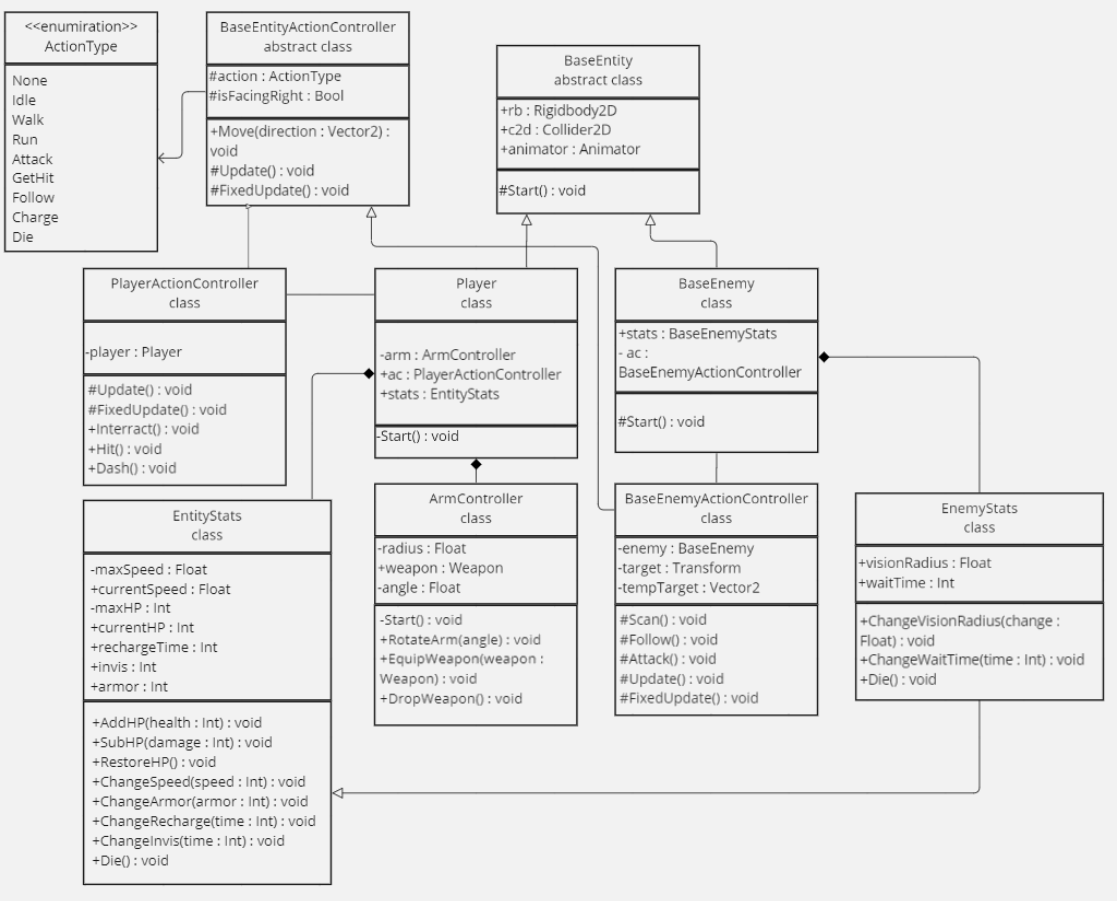


Рисунок 3.8 - Дерево класів істот

Далі додаю можливу зброю, яку гравець може отримати після вбивства всіх істот, також прописую функціонал самому персонажу для можливості зміни зброї. Також дороблюю меню, що залишилися.

Тепер залишилося додати UI та мобільне управління. Для візуалізації здоров'я знадобиться окремий клас, який зв'язуватиметься з характеристиками персонажа через клас Player. А керування для мобільних телефонів відбуватиметься за рахунок кнопок на екрані та підключеного до них класу MobileInputController.

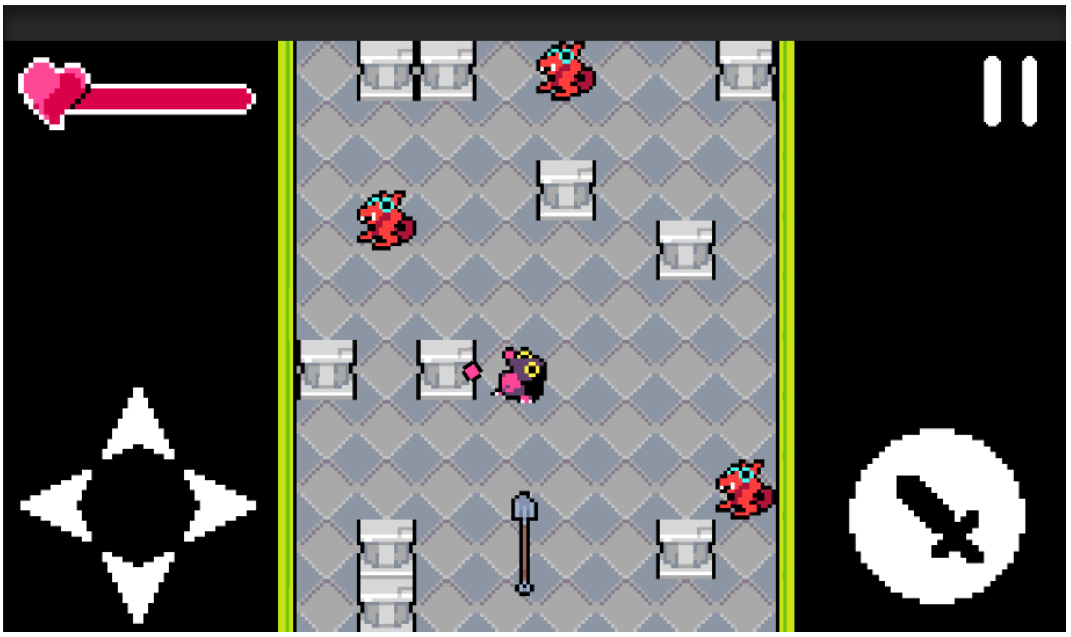


Рисунок 3.9 - Графічний інтерфейс мобільної гри

Для завершення створення додатку, його залишилось зібрати у працюючий білд для платформи Android у форматі apk. Після чого проект можна вважати виконаним та завершеним.

3.5 Висновки до розділу

У цьому розділу було створено дизайн документ із детальним описом всієї гри, створено багато схем та діаграм, по яким були розроблені елементи та частини гри. Був розроблений прототип для тестування ігрових механік та після затвердження створена вся графіка для проекту. Далі були розроблені меню, ігрові класи та алгоритми, за якими працює гра. Під кінець був створений мобільний інтерфейс, після чого розробка мобільної гри в жанрі екшн рогалик була завершена.

ВИСНОВКИ

Результатом виконаної дипломної роботи є комп'ютерна гра в жанрі екшн рогалик для платформи Android. У грі користувач попадає у випадково згенеровані рівні, де йому треба знищити всіх ворогів, після чого гравець потрапляє на наступний рівень. Особливістю гри в порівнянні з іншими аналогами є фокус ігрового процесу на близькому бої із ворогами. Для цього в користувача є можливість битися зброєю та ухилятися від атак ворогів. Після завершення рівня, гравцю надається вибір модифікатору, який може покращити його характеристики для більш успішної гри, після чого гравець може потрапити на новий рівень із більшою кількістю ворогів. При смерті персонажу гравець втрачає весь свій прогрес. Даний додаток розроблен для мобільних пристроїв на операційній системі Android.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Відеогра [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://uk.wikipedia.org/wiki/Відеогра
2. What are roguelike and roguelite video games? [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://www.makeuseof.com/what-are-roguelike-and-roguelite-video-games
3. Roguelike [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://en.wikipedia.org/wiki/Roguelike
4. Офіційний сайт гри “Nuclear Throne” [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://nuclearthrone.com/
5. Офіційний сайт гри “Hades”[Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://www.supergiantgames.com/games/hades/
6. Офіційний сайт гри “Soul Knight” [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: http://www.chillyroom.com/en
7. Game Engine [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://en.wikipedia.org/wiki/Game_engine
8. Офіційний сайт двигуна Unreal Engine [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://www.unrealengine.com/en-US
9. Офіційний сайт двигуна GameMaker Studio 2 [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://gamemaker.io/en/gamemaker
10. Офіційний сайт двигуна Unity [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://unity.com/
11. Офіційний сайт Microsoft Visual Studio [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://visualstudio.microsoft.com/ru/
12. Офіційний сайт Microsoft Visual Studio [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу https://www.aseprite.org/
13. Документація до ігрового двигуна Unity [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://docs.unity3d.com/Manual/index.html

14. Документація до розробки для платформи Android на ігровому двигуні Unity [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://docs.unity3d.com/Manual/android.html
15. User Interface In Games [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://ecampusontario.pressbooks.pub/gamedesigndevelopmenttextbook/chapter/user-interface/#:~:text=A%20user%20interface%20in%20games,-computer%20interactions%20(HCI).

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА ГРИ ЖАНРУ ЕКШН-РОГАЛИК “UNDERGROUNDER” ДЛЯ ПЛАТФОРМИ ANDROID НА ДВИГУНІ UNITY

Виконав студент 4 курсу
Групи ПД-41
Личаний Владислав Андріович
Керівник роботи
Доцент кафедри ІПЗ
д. ф. Дібрівний Олесь Андріович

Київ – 2022

ОСОБЛИВОСТІ ЖАНРУ ЕКШН-РОГАЛИК

- Циклічність ігрового процесу
- Випадкова генерація рівнів
- Наявність модифікаторів гравця
- Висока складність та вимогливість до реакції та швидкості гравця
- Високий темп ігри
- Відсутність збереження прогресу

АНАЛІЗ АНАЛОГІВ ІГОР В ЖАНРІ ЕКШН-РОГАЛИК

Наведені ігри	Nuclear Throne	Hades	Soul Knight
Платформи	Nintendo Switch, PlayStation 4, Xbox One, PlayStation 3, Windows, Linux, macOS	Nintendo Switch, PlayStation 4, PlayStation 5, Xbox One, macOS, Xbox Series X|S, Windows, Mac OS	Android, iOS, Nintendo Switch
Наявність мобільної версії	Ні	Ні	Так
Вартість копії	12$	25$	Умовно безкоштовна
Внутрішньоігрові покупки	Ні	Ні	Так
Складність геймплею	Висока	Висока	Середня
Наявність полегшеного режиму	Ні	Так	Ні
Наявність різних грабельних персонажів	Так	Ні	Так
Тип бою з ворогами	Дальній	Дальний	Дальній
Реіграбельність	Висока	Середня	Висока
Ігровий двигун	GameMaker Studio 2	Внутрішній ігровий двигун студії Supergiant Games	Unity

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи:** Покращення користувацького досвіду із внутрішньожанровим різномаїттям відеогр типу екшн-рогалик для мобільних платформ за рахунок розробки гри із фокусом на близький контакт із ворогами та піксельну стилізацію.
- **Об’єкт дослідження:** ігровий процес в жанрі екшн-рогалик.
- **Предмет дослідження:** програмний продукт жанрі екшн-рогалик для мобільної платформи Android.

ТЕХНІЧНЕ ЗАВДАННЯ

- Можливість контролювати персонажа, робити удари та ухялится
- Система здоров’я та характеристик персонажа і ворогів
- Генерація випадкових рівнів
- Перехід між екранами та рівнями
- Система модифікації характеристик персонажа

КОНЦЕПТ ГРИ “UNDERGROUNDER”

Жанр: екшн-рогалик

Платформа: Android

Цільова аудиторія: Хлопці 8-18 років

Стилізація: Контрастна та піксельна графіка

Місце дії: Підземне метро

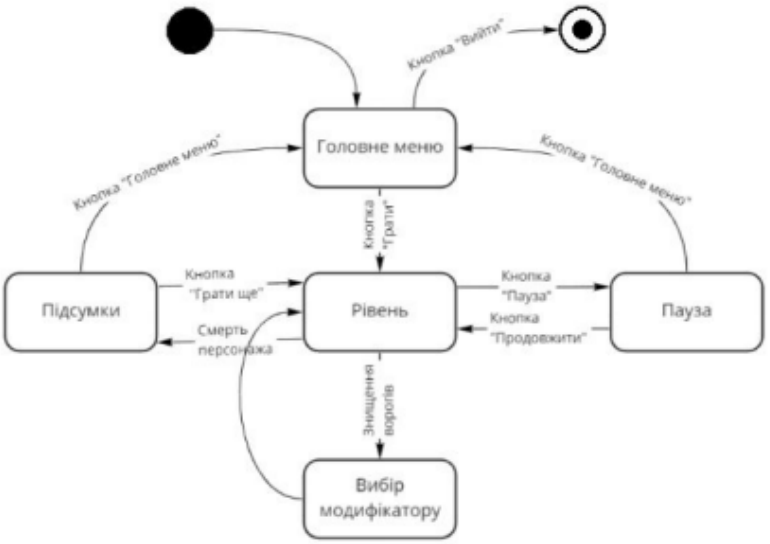
Головний герой: Кріт

Відмінність: Фокус на ближньому бої

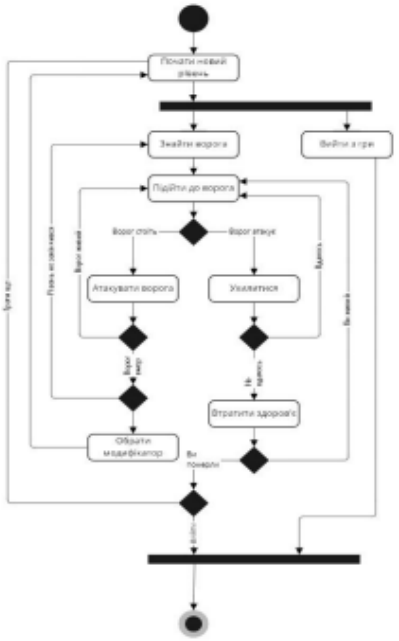
Двигун: Unity

ЗАСОБИ РОЗРОБКИ

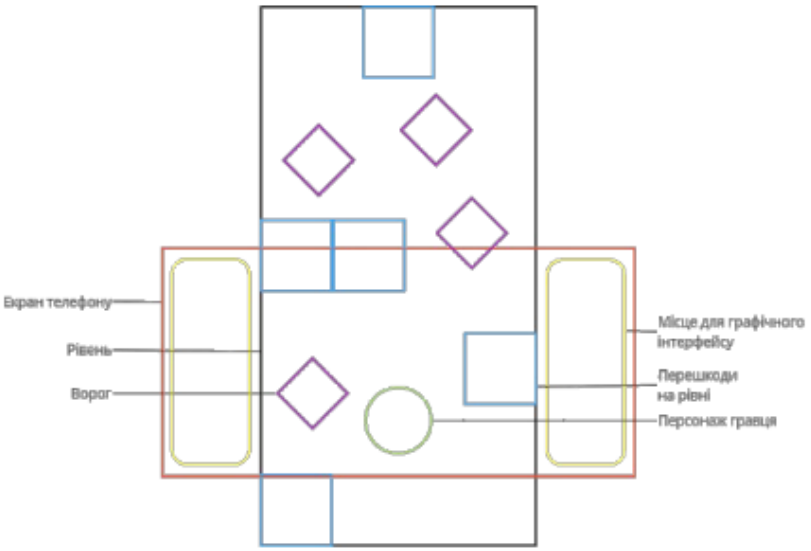
ГРАФ ДІАЛОГІВ



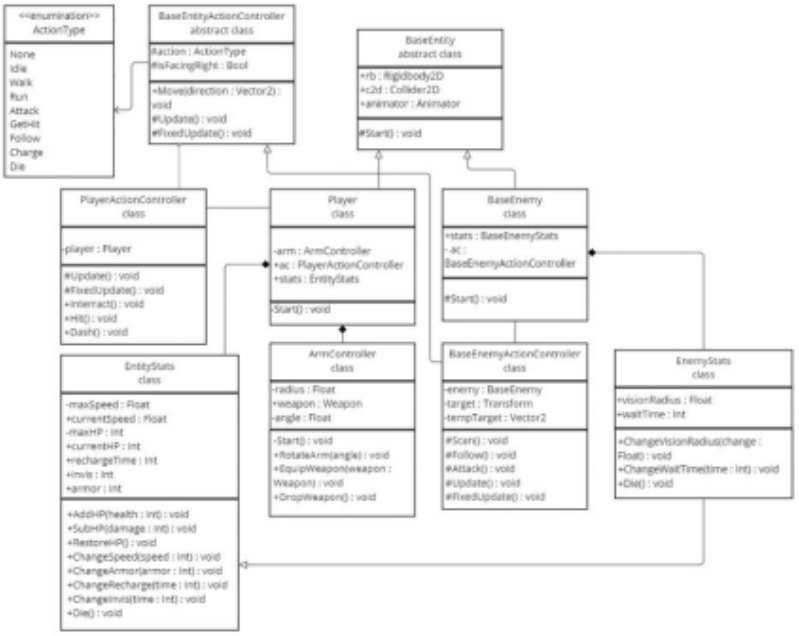
ДІАГРАМА ДІЯЛЬНОСТІ ГРАВЦЯ



МАКЕТ РІВНЯ

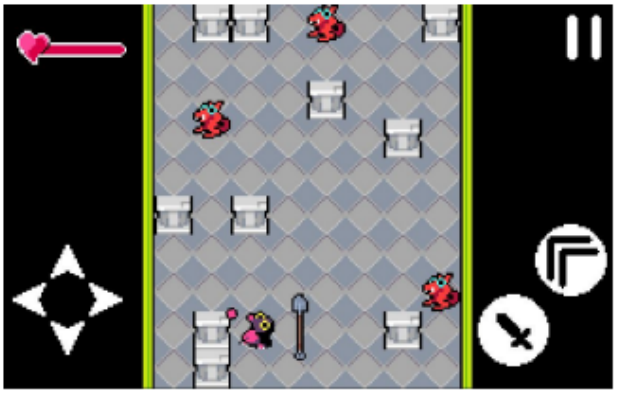


ДІАГРАМА КЛАСІВ ІСТОТ У ГРІ



ЕКРАННІ ФОРМИ

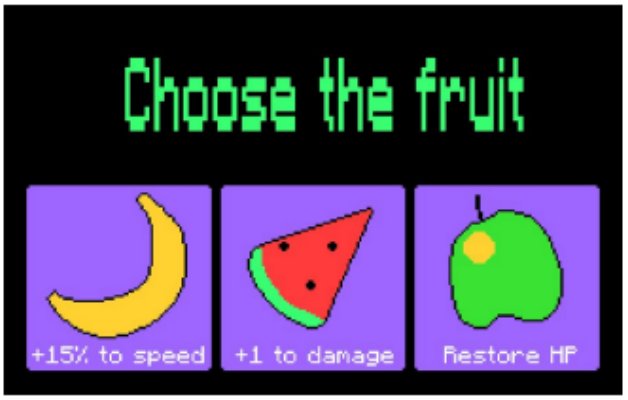
Головне меню



Рівень

ЕКРАННІ ФОРМИ

Смерть персонажа



Вибір модифікатору

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Личаний В.А. Екшн-рогалик як жанр відеоігор і його особливості: IV Міжнародна студентська конференція “Наука сьогодення: від досліджень до стратегічних рішень”, 13.06.2022р, м. Івано-Франківська

Личаний В.А. Засоби розробки мобільних ігор: IV Міжнародна студентська конференція “Наука сьогодення: від досліджень до стратегічних рішень”, 13.06.2022р, м. Івано-Франківська

ВИСНОВКИ

1. Проведено аналіз ігор в жанрі екшн-рогалик, визначені їх особливості, переваги і недоліки, серед останнього виділена відсутність ближнього бою у більшості проектів.
2. Розглянуті існуючі інструменти розробки відеоігор, в якості засобів розробки обран ігровий двигун Unity через безкоштовність, універсальність та можливості портування додатків на мобільні платформи.
3. Визначено технічне завдання гри, яке містить основні вимоги для мобільного додатка, створені діаграми, графи і схеми, які описують елементи гри.
6. Реалізована гра жанру екшн-роглаик “Undergrounder” для платформи Android із фокусом на ближній бій та піксельну графіку

ДЯКУЮ ЗА УВАГУ!