ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка мобільної гри жанру 2d scroll platform shooter з використанням мови програмування Python»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Антон КОМІСАРУК
(підпис)

Виконав: Здобувач вищої освіти групи ПД-41

Антон КОМІСАРУК

Керівник: Віталій ЗАЛИВА
доктор філософії (PhD)

Рецензент:

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Комісаруку Антону Володимировичу

1. Тема кваліфікаційної роботи: «Розробка мобільної гри жанру 2d scroll platform shooter з використанням мови програмування Python»
керівник кваліфікаційної роботи доктор філософії (PhD), старший викладач кафедри ІПЗ Віталій ЗАЛИВА,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» лютого 2024 р. № 36.
2. Строк подання кваліфікаційної роботи «28» травня 2024 р.
3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про жанр platform shooter, науково-технічна література з питань, пов’язаних з програмним забезпеченням щодо розробки ігор , технічна документація з описом бібліотек для розробки 2D ігор на Python та інструментів для створення мобільних додатків.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Огляд та аналіз існуючих методів та технологій розробки 2D ігор на Python.
2. Проектування гри жанру 2D scroll platform shooter.
3. Програмна реалізація та опис функціонування гри.
4. Тестування та оптимізація гри

5. Висновки

5. Перелік графічного матеріалу: *презентація*

1. Титульний слайд.
2. Мета, об’єкт та предмет дослідження
3. Технічне завдання.
4. Аналіз аналогів
5. Вимоги до програмного забезпечення.
6. Програмні засоби реалізації.
7. Діаграма використання.
8. Діаграма використання.
9. Діаграма класів.
10. Екранні форми.
11. Апробація результатів дослідження
12. Висновки

6. Дата видачі завдання «28» лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	28.02.-06.03.2024	
2	Аналіз та дослідження існуючих аналогів	07.03-13.03.2024	
3	Огляд існуючих технологій розробки 2D ігор та методів реалізації ігрових механік	14.03-20.03.2024	
4	Аналіз та вибір інструментів для розробки мобільного застосунку	21.03-28.03.2024	
5	Проектування архітектури мобільної гри	29.03-12.04.2024	
6	Розробка функціоналу гри	13.04-28.04.2024	
7	Оформлення роботи: вступ, висновки, реферат	29.04-05.05.2024	
8	Розробка демонстраційних матеріалів	06.05-12.05.2024	
9	Попередній захист роботи	13.05-31.05.2024	

Здобувач вищої освіти _____ Антон КОМІСАРУК
(підпис)

Керівник
кваліфікаційної роботи _____ Віталій ЗАЛИВА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 54 стор., 2 табл., 14 рис., 20 джерел.

Мета роботи – підвищення зацікавленості потенційних гравців можливістю створення власних рівнів та їх подальшого редагування.

Об'єкт дослідження – геймплей, що базується на проходженні платформних рівнів з можливістю їх створення та редагування.

Предмет дослідження – програмне забезпечення, що дозволяє створювати та редагувати рівні для самостійного проходження гри в жанрі platform shooter.

Короткий зміст роботи: В роботі проаналізовано технології та методи для розробки 2D ігор жанру scroll platform shooter мовою програмування Python. Проаналізовано інструментальні засоби для створення 2D ігор: Pygame, Godot, Unity. Розроблено архітектуру гри та програмно реалізовані ключові функціональні можливості, зокрема: ігровий цикл, рух персонажа, механіки стрибків та стрільби, інтеграція ворогів і перешкод. Проведено функціональне та модульне тестування гри. В роботі використано бібліотеку Pygame для розробки гри, Tkinter для створення користувацького інтерфейсу, Matplotlib для візуалізації даних.

Сферою використання гри є розвага користувачів та розвиток навичок програмування в процесі розробки ігор.

Наукова новизна дипломної роботи полягає у впровадженні інноваційних геймплейних механік, створенні унікального сценарію та використанні потужних функціональних можливостей Pygame для реалізації цього проекту.

У рамках дипломного проекту було проведено аналіз ринку комп'ютерних ігор у жанрі platform shooter. Було розглянуто переваги та недоліки різних програмних інструментів для розробки ігор. Також детально вивчено особливості розробки комп'ютерних ігор жанру platform shooter.

Комп'ютерна гра була розроблена за допомогою бібліотеки Pygame, а програмування ігрової логіки проводилося з використанням мови програмування

Python.

Розроблена комп'ютерна гра виходить за рамки простого академічного проекту. Її потенційна аудиторія широка: від казуальних гравців, які шукають нових вражень у класичному жанрі, до ретро-ентузіастів, що цінують автентичний досвід scroll shooter. Більше того, ця гра стає цінним освітнім інструментом. Для студентів, які вивчають програмування і розробку ігор, вона є живим прикладом того, як теоретичні концепції - від алгоритмів пошуку шляху до управління ресурсами - реалізуються в реальному ігровому проекті. Код гри, документований за найвищими стандартами, може використовуватися в курсах з Python, ігрової розробки, навіть в уроках дизайну рівнів.

ЗМІСТ

ВСТУП

В сучасному світі, технології продовжують швидко розвиватися, змінюючи наше життя та повсякденну діяльність. Однією з галузей, яка зазнала значного впливу технологій, є індустрія відеоігор. Завдяки появі мобільних пристроїв, таких як смартфони та планшети, відеоігри стали доступнішими для широкої аудиторії, а мобільні ігри - одним з найпопулярніших видів розваг.

За даними досліджень, кількість користувачів мобільних ігор у всьому світі перевищила у 2022 році близько 90 млрд мобільних ігор, а ринок мобільних ігор оцінюється вже більше 100 мільярда доларів США. Ці цифри свідчать про те, що створення мобільних ігор є не тільки актуальним, але й перспективним напрямком розвитку.

Однією з популярних категорій мобільних ігор є platformer. Ці ігри поєднують в собі елементи платформерів та шутерів, забезпечуючи гравцям динамічний та захоплюючий ігровий процес. Крім того, завдяки використанню 2D графіки, ці ігри можуть бути легко адаптовані для мобільних пристроїв та мати невеликий розмір, що є важливим фактором для користувачів.

Враховуючи вищезазначене, створення мобільної гри 2D scroll platformer shooter мовою програмування Python є актуальним та перспективним напрямком. Python - це популярна та широко використовувана мова програмування, яка має багато переваг для розробки ігор. Зокрема, Python має простий та зрозумілий синтаксис, що дозволяє швидко та ефективно розробляти ігри. Крім того, Python має велику кількість бібліотек та фреймворків для розробки ігор, таких як Pygame, Panda3D та Cocos2d, що спрощують процес розробки та дозволяють зосередитися на творчому аспекті.

Цей проект дозволить не тільки розвинути навички програмування на Python, але й отримати досвід у створенні мобільних ігор, що є важливим для майбутньої кар'єри в галузі розробки програмного забезпечення. Крім того, цей

проект може стати основою для подальших досліджень та розробок в галузі мобільних ігор.

Об'єкт дослідження – геймплей, що базується на проходженні платформних рівнів з можливістю їх створення та редагування.

Предмет дослідження – програмне забезпечення, що дозволяє створювати та редагувати рівні для самостійного проходження гри.програмне забезпечення, що дозволяє створювати та редагувати рівні для самостійного проходження гри в жанрі platform shooter.

Мета роботи – підвищення зацікавленості потенційних гравців можливістю створення власних рівнів та їх подальшого редагування.

Новизна мобільної гри 2D scroll platformer shooter, яка була створена в ході роботи, полягає в тому, що гравці мають можливість створювати та редагувати власні рівні у грі. Ця особливість відрізняє цю гру від інших ігор в цій категорії та робить її більш привабливою для гравців.

Даючи гравцям можливість створювати власні рівні, в ході роботи було збільшено тривалість та цікавість гри, а також залучено гравців до творчого процесу. Гравці можуть проявляти свою креативність та уяву, створюючи унікальні та цікаві рівні, які потім можуть бути розділені з іншими гравцями.

Крім того, можливість редагування рівнів дозволить гравцям вносити зміни та покращення в існуючі рівні, щоб зробити їх більш складними та цікавими. Це додасть грі більшої глибини та реіграбельності, оскільки гравці можуть проходити одні та ті самі рівні знову та знову, але з кожним разом вони будуть відрізнятися.

1. ТЕОРЕТИЧНІ АСПЕКТИ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Мобільні відеоігри

Мобільні ігри це феномен цифрового розважального середовища.

У сучасну еру, де смартфони стали невід'ємною частиною нашого повсякденного життя, мобільні ігри посіли чільне місце в індустрії цифрових розваг. Ці компактні, але потужні платформи для гейміфікації не лише змінили спосіб взаємодії людей з відеоіграми, але й трансформували саму природу ігрової індустрії.

Мобільні ігри - це відеоігри, розроблені спеціально для смартфонів, планшетів та інших портативних пристроїв. На відміну від традиційних консольних або комп'ютерних ігор, мобільні ігри оптимізовані для сенсорних екранів, використовують вбудовані функції пристроїв (як-от акселерометр або GPS) і часто вимагають менше обчислювальної потужності. Це дозволяє створювати ігри, які можна легко запускати та грати в будь-якому місці та в будь-який час.

Жанрова палітра мобільних ігор надзвичайно різноманітна. Вона охоплює все: від простих головоломок і аркад (наприклад, "Candy Crush Saga" або "Angry Birds") до складних стратегій реального часу ("Clash of Clans"), рольових ігор ("Genshin Impact") і навіть багатокористувацьких онлайн-баталій ("PUBG Mobile").

Чому ж мобільні ігри стали такими популярними ?

Доступність: З поширенням смартфонів майже кожен має доступ до мобільних ігор. За даними Newzoo, у 2023 році кількість власників смартфонів у світі перевищила 4,6 мільярда.

Простота використання: Інтуїтивні сенсорні інтерфейси роблять мобільні ігри доступними навіть для тих, хто ніколи раніше не грав у відеоігри.

Гнучкість часу: В сучасному швидкому ритмі життя не завжди є час для довгих ігрових сесій. Мобільні ігри пропонують короткі, але захопливі ігрові сеанси, ідеальні для проведення часу в черзі, під час перерви або в транспорті.

Різноманітність: Ігрові магазини, такі як Google Play Store і Apple App Store, пропонують сотні тисяч ігор на будь-який смак і бюджет.

Безкоштовна модель: Багато мобільних ігор використовують модель "freemium" - безкоштовне завантаження з можливістю внутрішньоігрових покупок. Це знижує бар'єр входу, дозволяючи гравцям спробувати гру без початкових витрат.

Соціальна інтеграція: Мобільні ігри часто інтегровані з соціальними мережами, дозволяючи гравцям змагатися з друзями, обмінюватися досягненнями або просити про допомогу, підсилюючи соціальну взаємодію.

Постійні оновлення: Розробники регулярно додають новий контент, події та функції, утримуючи гравців зацікавленими протягом тривалого часу.

Популярність мобільних ігор має значний економічний вплив. За даними App Annie, у 2020 році, під час пандемії COVID-19, загальні споживчі витрати в магазинах мобільних додатків досягли $143 мільярдів, з яких ігри склали понад 70%. Навіть у пост-пандемічний період ця тенденція зберігається. У першому кварталі 2023 року витрати на мобільні ігри все ще перевищували $20 мільярдів.

Такий успіх призвів до того, що традиційні ігрові гіганти, такі як Nintendo ("Super Mario Run"), Activision Blizzard ("Call of Duty: Mobile") і Electronic Arts ("FIFA Mobile"), активно входять на мобільний ринок. Крім того, з'явились нові титани, як-от Supercell (Фінляндія) і miHoYo (Китай), які спочатку зосередились на мобільних іграх.

Мобільні ігри - це не просто тренд, а фундаментальна зміна парадигми в ігровій індустрії. Поєднуючи доступність, зручність і різноманітність, вони залучають величезну і різноманітну аудиторію, виходячи за межі традиційних демографічних груп геймерів. З постійним удосконаленням мобільних технологій та появою 5G-мереж, майбутнє мобільних ігор виглядає ще яскравішим, обіцяючи

більш складні, візуально вражаючі та соціально пов'язані ігрові досвіди. У світі, де смартфон завжди під рукою, мобільні ігри стали універсальною формою цифрових розваг.

1.2 Жанри

Жанри мобільних ігор - це різні категорії, на які можна розділити мобільні відеоігри залежно від їхніх характеристик та особливостей. Жанри допомагають гравцям знайти ігри, які їм цікаві, та допомагають розробникам програмного забезпечення визначити, якого типу ігри найбільш популярні серед гравців.

Існує багато різних жанрів мобільних ігор, але деякі з них є більш популярними та поширеними, ніж інші. Ось деякі з найпопулярніших жанрів мобільних ігор:

- Аркадні ігри: це прості та захоплюючі ігри, які часто містять декілька рівнів та вимагають швидких рефлексів та навичок керування.
- Гоночні ігри: це ігри, в яких гравці керують транспортними засобами та змагаються з іншими гравцями або з часом.
- Платформери: це ігри, в яких гравці керують персонажем, який повинен долати перешкоди та ворогів, щоб дістатися до кінця рівня.
- Шутери: це ігри, в яких гравці керують персонажем, який стріляє в ворогів або цілі.
- Стратегічні ігри: це ігри, в яких гравці повинні розробляти та впроваджувати стратегію, щоб досягти певної мети або перемогти суперника.
- Рольові ігри (RPG): це ігри, в яких гравці керують персонажем, який розвивається та вдосконалюється протягом гри, виконуючи завдання та змагаючись з ворогами.
- Головоломки: це ігри, в яких гравці повинні розв'язувати загадки або головоломки, щоб просунутися вперед.

Програма дипломної роботи це 2D scroll platformer shooter є гібридом двох жанрів: платформерів та шутерів. Цей гібридний жанр є популярним серед гравців, оскільки він поєднує в собі елементи двох популярних жанрів та забезпечує динамічний та захоплюючий ігровий процес.

Загалом, жанри мобільних ігор є важливим аспектом розробки та просування мобільних відеоігор. Вони допомагають гравцям знайти ігри, які їм цікаві, та допомагають розробникам програмного забезпечення визначити, якого типу ігри найбільш популярні серед гравців.

1.3 Платформи

Платформи для ігор - це різні пристрої та системи, на яких можна грати в відеоігри. Кожна платформа для ігор має свої особливості, переваги та недоліки, а також свою власну аудиторію.

Платформи можна розділити на декілька категорій, ось їх список та опис:

Персональні комп'ютери (PC): це найбільш універсальна та поширена платформа для ігор. На персональних комп'ютерах можна грати в широкий спектр ігор, від простих аркадних ігор до складних стратегічних та рольових ігор. Персональні комп'ютери мають більші можливості для гри в складні та ресурсомісткі ігри, ніж інші платформи, але вони також можуть бути більш дорогими та менш доступними.

Консолі: це спеціалізовані пристрої, призначені для гри в відеоігри. Найпопулярнішими консолями є PlayStation, Xbox та Nintendo. Консолі мають свої власні ексклюзивні ігри, які не можна грати на інших платформах. Консолі також мають більші можливості для гри в складні та ресурсомісткі ігри, ніж мобільні пристрої, але вони є більш дорогими та менш доступними.

Мобільні пристрої: це смартфони та планшети, які можна використовувати для гри в мобільні відеоігри. Мобільні пристрої є найбільш доступною та поширеною платформою для ігор, оскільки більшість людей мають мобільні

пристрої. Мобільні пристрої мають менші можливості для гри в складні та ресурсомісткі ігри, ніж консолі та персональні комп'ютери, але вони є більш зручними та доступними для гри в будь-який час та в будь-якому місці.

Браузерні ігри: це окрема платформа для ігор, яка дозволяє грати в ігри безпосередньо у браузері. Вони є зручними та доступними, оскільки не вимагають встановлення додаткового програмного забезпечення. На жаль мають менші можливості для гри в складні та ресурсомісткі ігри, ніж інші платформи, але вони є зручними та доступними для простих та легких ігор.

Віртуальна реальність (VR): це нова та перспективна платформа для ігор, яка дозволяє гравцям занурюватися в віртуальний світ та взаємодіяти з ним. Відповідно до прогнозів, ринок VR-ігор буде швидко зростати в найближчі роки. VR-ігри мають більші можливості для гри в складні та ресурсомісткі ігри, ніж інші платформи, але вони також можуть бути більш дорогими та менш доступними.

Кожна платформа для ігор має свої особливості, переваги та недоліки, а також свою власну аудиторію. Вибір платформи для гри залежить від багатьох факторів, включаючи доступність, зручність, вартість та особливості платформи.

1.4 Порівняння існуючих аналогів

Жанр платформер шутер (platformer shooter) є поєднанням двох популярних жанрів відеоігор: платформерів (platformers) та шутерів (shooters).

Платформери - це відеоігри, в яких гравець керує персонажем, який повинен пересуватися по рівнях, стрибати на платформи, ухилятися від перешкод та ворогів, та виконувати інші дії, щоб досягти кінця рівня або виконати певне завдання.

Шутери - це відеоігри, в яких гравець керує персонажем, який повинен стріляти в ворогів, щоб вижити та виконати певне завдання.

Жанр платформер шутер поєднує в собі елементи обох жанрів, створюючи цікавий та захоплюючий ігровий процес. Гравець повинен не тільки стрибати на платформи та ухилятися від перешкод, але також стріляти в ворогів, щоб вижити та виконати завдання.

Жанр платформер шутер є популярним серед гравців, оскільки він забезпечує цікавий та захоплюючий ігровий процес, а також надає можливість проявляти свою спритність, реакцію, та стратегічне мислення.

Історія жанру платформер шутер починається з ранніх відеоігор, таких як "Space Invaders" (1978) та "Pac-Man" (1980), в яких гравець повинен був стріляти в ворогів та ухилятися від них. Однак, першою справжньою грою в жанрі платформер шутер вважається "Contra" (1987), в якій гравець повинен був стріляти в ворогів, стрибати на платформи, та виконувати інші дії, щоб вижити та виконати завдання.

З тих пір жанр платформер шутер розвивався та еволюціонував, створюючи такі популярні відеоігри, як "Metal Slug" (1996), "Cuphead" (2017), та "Hollow Knight" (2017).

Contra та Strike Force Heroes - це популярні відеоігри в жанрі платформер шутер, які мають багато спільних рис та переваг.

Contra - це відеогра, розроблена та випущена компанією Konami в 1987 році. Гравець керує солдатом, який повинен стріляти в ворогів, стрибати на платформи, та виконувати інші дії, щоб вижити та виконати завдання. Гра має просту, але захоплюючу механіку, а також різноманітні рівні та ворогів.

Рис. 1.1 Знімок з гри Contra

Strike Force Heroes - це відеогра, розроблена та випущена компанією Sky9 Games в 2012 році. Гравець керує солдатом, який повинен стріляти в ворогів, стрибати на платформи, та виконувати інші дії, щоб вижити та виконати завдання. Гра має більш складну механіку, ніж Contra, включаючи систему класів, систему зброї, та систему нагород. Також гра має більш сучасну графіку та звукові ефекти.

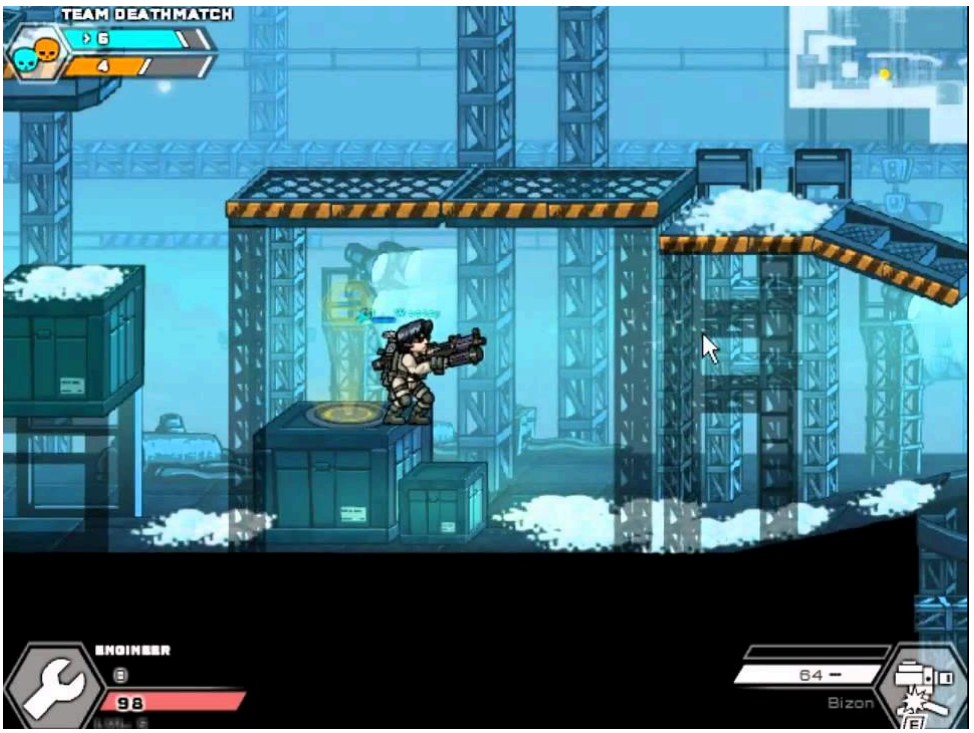


Рис.1.2 Знімок з гри Strike Force Heroes

Обидві гри мають багато спільних рис та переваг, включаючи:

Захоплюючий ігровий процес: обидві гри мають просту, але захоплюючу механіку, яка забезпечує цікавий та захоплюючий ігровий процес.

Різноманітність рівнів та ворогів: обидві гри мають різноманітні рівні та ворогів, щоб забезпечити різноманітність та реіграбельність.

Система нагород та прогресу: обидві гри мають систему нагород та прогресу, щоб заохочувати гравця досягати кращих результатів та продовжувати гру.

Проста графіка та звукові ефекти: обидві гри мають просту, але яскраву графіку та звукові ефекти, щоб забезпечити повне занурення у світ гри.

Contra та Strike Force Heroes - це популярні відеоігри в жанрі платформер шутер, які мають багато спільних рис та переваг. Ці ігри є гарними прикладами для натхнення та вивчення при розробці вашої мобільної гри в жанрі платформер шутер.

Таблиця 1.1

Порівняльна таблиця існуючих аналогів

Характеристика	Strike Force Heroes	Contra	Nolum`s mission
Платформа	Веб-браузер	ПК, ігрові консолі	ПК, Мобільні пристрої
Мова програмування	ActionScript 3.0	Ассемблер, C	Python
Кількість активних гравців	1	1-2	1
Можливість кастомізації рівнів	-	-	+
Наявність бонусів	+	+	+
Індикатори характеристик	+	-	+
Реіграбельність	+	-	+

1.5 Аналіз Цільової аудиторії

Аналіз цільової аудиторії - це процес визначення та опису характеристик, вподобань, потреб та поведінки людей, які є потенційними користувачами або гравцями програмного забезпечення або відеоігри. Аналіз цільової аудиторії є важливим для розробки програмного забезпечення та відеоігор, оскільки він допомагає розробникам програмного забезпечення та дизайнерам відеоігор розуміти, для кого вони створюють свої продукти та як вони можуть задовольнити потреби та уподобання своєї цільової аудиторії.

Аналіз цільової аудиторії для ігор, включаючи мобільні відеоігри жанру платформер шутер, може включати такі кроки:

Визначення цільової аудиторії: розробники програмного забезпечення та дизайнери відеоігор повинні визначити, для кого вони створюють свої продукти, включаючи вік, стать, освіту, досвід, інтереси, уподобання, поведінку та інші характеристики.

Дослідження цільової аудиторії: розробники програмного забезпечення та дизайнери відеоігор повинні провести дослідження, щоб дізнатися більше про свою цільову аудиторію, включаючи опитування, фокус-групи, аналіз даних соціальних мереж, та інші методи дослідження.

Аналіз даних: розробники програмного забезпечення та дизайнери відеоігор повинні аналізувати дані, зібрані під час дослідження, щоб визначити ключові характеристики, вподобання, потреби та поведінку своєї цільової аудиторії.

Створення персонажів цільової аудиторії: розробники програмного забезпечення та дизайнери відеоігор можуть створити персонажів цільової аудиторії, які представляють ключові характеристики, вподобання, потреби та поведінку своєї цільової аудиторії, щоб допомогти їм розуміти та спілкуватися з своєю цільовою аудиторією.

Тестування та зворотний зв'язок: розробники програмного забезпечення та дизайнери відеоігор повинні тестувати свої продукти з цільовою аудиторією та

збирати зворотний зв'язок, щоб визначити, чи задовольняють їхні продукти потреби та уподобання своєї цільової аудиторії, та чи є потенціал для покращення.

Також до аналізу цільової аудиторії можна віднести і додаткові характеристики, такі як загальні вподобання, потреби та поведінку:

Вік: мобільні відеоігри жанру платформер шутер можуть бути популярними серед людей різного віку, але найбільш імовірною цільовою аудиторією є підлітки та молоді люди віком від 13 до 30 років.

Освіта: мобільні відеоігри жанру платформер шутер можуть бути популярними серед людей з різним рівнем освіти, але найбільш імовірною цільовою аудиторією є люди з середньою або вищою освітою.

Досвід: мобільні відеоігри жанру платформер шутер можуть бути популярними серед людей з різним рівнем досвіду в іграх, але найбільш імовірною цільовою аудиторією є люди з помірним або низьким рівнем досвіду в іграх.

Інтереси: мобільні відеоігри жанру платформер шутер можуть бути популярними серед людей з різними інтересами, але найбільш імовірною цільовою аудиторією є люди, які цікавляться дією, пригодами, фантастикою, та науковою фантастикою.

Вподобання: мобільні відеоігри жанру платформер шутер можуть бути популярними серед людей з різними вподобаннями, але найбільш імовірною цільовою аудиторією є люди, які цікавляться швидкими, динамічними, та захоплюючими іграми з простим та інтуїтивно зрозумілим керуванням.

Потреби: мобільні відеоігри жанру платформер шутер можуть задовольняти такі потреби людей, як розвага, відпочинок, стрес-менеджмент, та самовираження.

Поведінка: мобільні відеоігри жанру платформер шутер можуть бути популярними серед людей з різними типами поведінки, але найбільш імовірною цільовою аудиторією є люди, які люблять грати в ігри в будь-який час та в будь-якому місці, та люди, які люблять змагатися з іншими гравцями або з часом.

Визначення цільової аудиторії: аналіз цільової аудиторії базується на загальних тенденціях у жанрі, враховуючи історичний контекст і сучасні реалії ринку.

1. Ностальгуючі геймери (35-50 років)

Опис: Це гравці, які виросли в 80-і та 90-і роки, золоту епоху аркадних ігор і консолей типу NES, SNES і SEGA Genesis.

Мотивація: Ностальгія за класичними іграми, такими як "Contra", "Metal Slug", "Mega Man". Вони шукають автентичний досвід, який нагадує їм про юність.

Особливості: Цінують точне відтворення механік старої школи - складні стрибки, патерни ворогів, колекціонування зброї.

2. Хардкорні платформери (18-34 роки)

Опис: Ентузіасти складних платформерів і "метроїдваній", такі як фанати "Super Meat Boy", "Celeste", "Hollow Knight".

Мотивація: Прагнуть викликів, насолоджуються вдосконаленням своїх навичок. Шукають ігри, що вимагають точності й майстерності.

Особливості: Цінують чудово розроблені рівні, що вимагають точного таймингу, різноманітні механіки руху (подвійні стрибки, біг по стінах).

3. Інді-геймери (25-40 років)

Опис: Цінителі незалежних ігор, слідкують за сценою через платформи як itch.io, Steam, стежать за розробниками в Twitter.

Мотивація: Шукають унікальні, інноваційні ігри, які відходять від мейнстріму. Цінують авторський стиль і експериментальний геймплей.

Особливості: Приваблює оригінальна естетика, експериментальні механіки, нелінійні наративи.

4. Молоді дорослі-казуали (25-35 років)

Опис: Зайняті професіонали, які грають у перервах між роботою або ввечері для розслаблення.

Мотивація: Шукають швидку розвагу та викид адреналіну, люблять почуття прогресу.

Особливості: Цінують короткі ігрові сесії, чіткі цілі. Не люблять високої складності, що викликає фрустрацію.

5. Молоде покоління (13-17 років)

Опис: Підлітки, які відкривають для себе різні жанри ігор. Багато хто з них знайомиться з класичними жанрами через ремейки або інді-ігри.

Мотивація: Експериментують, шукають "свій" жанр. Сильно підпадають під вплив стримерів і YouTube-блогерів.

Особливості: Цінують можливість самовираження, кастомізацію персонажа. Люблять ділитися досягненнями в соціальних мережах.

6. Студенти-програмісти (18-25 років)

Опис: Вивчають програмування, особливо Python. Цікавляться розробкою ігор як хобі або майбутньою кар'єрою.

Мотивація: Шукають проекти для навчання, хочуть зрозуміти, як працюють ігри "під капотом".

Особливості: Цінують доступний код, хороші коментарі. Люблять ігри, які демонструють цікаві технічні трюки.

7. Творча спільнота (різний вік)

Опис: Художники, музиканти, дизайнери рівнів, які цікавляться індустрією ігор.

Мотивація: Шукають платформу для своєї творчості, хочуть бути частиною ігрового проекту.

Особливості: Цінують ігри з сильною естетикою, яскравими персонажами, захоплюючими світами.

Мобільна гра жанру платформер шутер може бути популярною серед людей різного віку, статі, освіти, досвіду, інтересів, уподобань, потреб та поведінки. Однак, найбільш імовірною цільовою аудиторією для вашої гри можуть бути підлітки та молоді люди віком від 13 до 30 років, чоловіки, люди з середньою або

вищою освітою, люди з помірним або високим рівнем досвіду в іграх, люди, які цікавляться дією, пригодами, фантастикою, та науковою фантастикою, люди, які цікавляться швидкими, динамічними, та захоплюючими іграми з простим та інтуїтивно зрозумілим керуванням, люди, які люблять грати в ігри в будь-який час та в будь-якому місці, та люди, які люблять змагатися з іншими гравцями або з часом.

2. АНАЛІЗ ТА ВИБІР ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1 Мова програмування Python

У світі розробки ігор, де традиційно панують такі мови, як C++ і C#, вибір Python для створення 2D scroll platform shooter може здатися незвичайним. Однак це рішення не лише логічне, але й стратегічно обґрунтоване, особливо в контексті освітньої місії проекту та його потенційної адаптації для мобільних платформ.

Python відомий своїм чистим і зрозумілим синтаксисом. У проекті, де розробник маніпулює складними механіками - від точного таймінґу стрибків до балістики зброї - код повинен бути максимально читабельним. Python дозволяє висловлювати алгоритми майже так, як вони були б написані на псевдокоді. Наприклад, код для механіки подвійного стрибка настільки виразний, що навіть непрограміст може його зрозуміти. Ця прозорість коду не лише полегшує розробку та підтримку гри, але й робить її чудовим навчальним інструментом, демістифікуючи процес створення ігор для студентів.

Ігровий дизайн - це процес постійних ітерацій. У розробці платформера кожен елемент, від висоти стрибка до інерції бігу, потребує тонкого налаштування. Python з його динамічною типізацією і багатою стандартною бібліотекою дозволяє швидко прототипувати і тестувати ідеї. Розробник може зосередитися на геймплеї, а не на боротьбі з компілятором. За один день можна перепробувати десяток варіантів механіки стрільби, що значно прискорює творчий процес.

Вибір Pygame для цього проекту також виявився геніальним. На відміну від громіздких движків, Pygame - це бібліотека низького рівня, що надає прямий доступ до графіки, звуку та введення. У платформерах, де кожен піксель і мілісекунда мають значення, Pygame дає розробнику точний контроль над усіма аспектами гри. Відсутність непотрібних функцій, які споживають ресурси, критично важлива для підтримки стабільних 60 кадрів на секунду на різних пристроях. Більше того, робота з Pygame дозволяє глибоко зрозуміти, як

працюють ігри зсередини. Це не "чорна скринька" движка, а власний код розробника керує кожним спрайтом і колізією.

Унікальність Python не обмежується його синтаксисом. Філософія "Дзен Python", яка цінує простоту та явність над складністю, резонує з дизайном класичних платформерів, де ясні правила і чисті механіки створюють захоплюючий досвід. Багата екосистема Python також грає на користь цього вибору. NumPy і SciPy використовуються для складних фізичних обчислень, таких як траєкторії снарядів. Matplotlib, який розробник згадував для візуалізації даних, стає в пригоді для аналізу геймплею - від теплокарт смертей гравців до графіків швидкості проходження рівнів. Навіть scikit-learn може бути застосований для створення адаптивної складності, навчаючи ШІ на даних гравців.

Python також відкриває двері до більш просунутих технік розробки ігор. Його здібності до метапрограмування дозволяють коду модифікувати себе під час виконання. Це відкриває захоплюючі можливості для процедурної генерації рівнів, де код може динамічно створювати нові класи ворогів або перешкод. Інтерактивні середовища Python, такі як REPL і Jupyter, стають секретною зброєю розробника, дозволяючи налагоджувати фізику стрибків у реальному часі, змінюючи параметри на льоту.

Коли справа доходить до мобільної розробки, Python знову демонструє свою універсальність. Хоча для ПК-версії платформера використовується Pygame, для мобільних платформ існують потужні фреймворки, такі як Kivy і BeeWare. Kivy, з його відкритим кодом, ідеально підходить для портування гри на Android та iOS, зберігаючи при цьому багатосенсорну функціональність. BeeWare йде ще далі, дозволяючи використовувати нативні віджети, що робить гру природною для кожної платформи.

Навіть якщо фінальна версія мобільної гри буде написана на Java для Android або Swift для iOS, Python залишається неперевершеним інструментом для швидкого прототипування. Розробник використав Python і Pygame для створення прототипу, перевіряючи всі механіки до найдрібніших деталей. Цей підхід

заощадив місяці розробки перед складним процесом портування на мобільні платформи.

У сучасних мобільних іграх онлайн-компоненти стали майже стандартом, і тут Python знову виходить на перший план. Django і Flask використовуються для створення надійних API, що обслуговують таблиці лідерів, профілі гравців і внутрішньоігрові магазини. Для багатокористувацьких режимів у реальному часі, які стають все популярнішими навіть у платформерах, бібліотека Tornado забезпечує швидкий обмін даними.

Аналітика гравців - ще одна сфера, де Python сяє. З його потужними інструментами для аналізу даних (pandas, NumPy) і машинного навчання (scikit-learn, TensorFlow), розробники можуть глибоко аналізувати поведінку гравців. Це дозволяє оптимізувати монетизацію, яка є ключовим фактором у "freemium" моделі більшості мобільних ігор, і проводити A/B-тестування нових функцій.

Python також інтегрується з іншими інструментами розробки ігор. Blender, з його потужним Python API, може бути використаний для створення і анімації 2D-спрайтів для платформера. Tiled, популярний редактор рівнів, підтримує сценарії Python, дозволяючи автоматизувати дизайн рівнів, які потім безпосередньо імпортуються в гру.

Вибір Python для розробки 2D scroll platform shooter є не лише обґрунтованим, але й інноваційним рішенням. У світі, де мобільні технології домінують, Python пропонує унікальний баланс простоти, потужності та гнучкості. Він дозволяє розробнику зосередитися на тому, що дійсно важливо - створенні захоплюючого ігрового досвіду, будь то на ПК або в кишені гравця.

Більше того, використовуючи Python, розробник робить більше, ніж просто розважальний продукт. Він створює потужний освітній інструмент, що розкриває таємниці процесу розробки ігор. У сучасному світі, де інтерес до розробки ігор серед молоді стрімко зростає, цей проект стає не лише джерелом розваги, але й

мостом, що з'єднує мрії з реальністю. Кожен рядок Python-коду - це урок, кожен спрайт - це можливість навчання.

У глобальному масштабі вплив Python і таких проектів, як цей платформер, виходить далеко за межі одного додатка. Python, з його філософією простоти та потужності, стає каталізатором інновацій у ігровій індустрії. Він надихає розробників - від студентів до професіоналів - експериментувати з новими ідеями, не боячись технічних бар'єрів.

У епоху, коли мобільні ігри переросли з простої розваги в культурний феномен, вибір Python для розробки 2D scroll platform shooter є свідченням глибокого розуміння сучасних тенденцій. Цей вибір відображає не лише технічну компетентність, але й стратегічне бачення, де доступність, навчання та інновації переплітаються, формуючи майбутнє ігрової індустрії. У світі, де смартфон завжди під рукою, а Python - в серці творчого процесу, межі між навчанням і розвагою, між мобільним та традиційним геймінгом, стираються, відкриваючи нові горизонти для творців та гравців.

2.2 Середовище розробки Visual Studio

Visual Studio Code (або ж VS Code) - це безкоштовний та відкритий текстовий редактор та середовище розробки, розроблене Microsoft. VS Code є популярним серед розробників програмного забезпечення завдяки своїй гнучкості, продуктивності та великій кількості додатків та розширень.

Visual Studio від Microsoft вже понад два десятиліття залишається одним з найпотужніших і найбільш шанованих інструментів у цій галузі. У контексті розробки 2D scroll platform shooter на Python, як у вашому проекті, Visual Studio пропонує унікальний набір можливостей, що підвищують продуктивність та якість розробки.

Visual Studio, вперше представлена в 1997 році, пройшла довгий шлях еволюції, постійно адаптуючись до мінливих потреб розробників. Спочатку

орієнтована на розробку під Windows з використанням Visual Basic і C++, сьогодні вона підтримує широкий спектр мов і платформ. Цей прогрес відображає філософію Microsoft щодо інклюзивності та адаптивності - якості, надзвичайно важливі в динамічній індустрії відеоігор.

Хоча Python не є "рідною" мовою для Visual Studio, як C# або C++, інтеграція Python стала одним з найбільш визначних покращень останніх років. З випуском розширення Python Tools for Visual Studio (PTVS) та подальшою глибокою інтеграцією в IDE, Microsoft чітко визнала зростаючу популярність Python серед розробників, особливо в таких сферах, як наукові обчислення, машинне навчання і, звичайно ж, розробка ігор.

Ось основні переваги VS Code:

Гнучкість: VSCode є гнучким та налаштовуваним середовищем розробки, яке дозволяє розробникам програмного забезпечення налаштовувати його відповідно до своїх потреб та вподобань.

Продуктивність: VS Code має багато функцій, які допомагають розробникам програмного забезпечення бути більш продуктивними, таких як автодоповнення коду, відлагодження, контроль версій, та інтеграція з іншими інструментами розробки.

Додатки та розширення: VS Code має велику кількість додатків та розширень, які можна встановити для розширення можливостей середовища розробки. Ці додатки та розширення можуть бути корисними для розробки програмного забезпечення в різних мовах програмування та платформах.

Крос-платформність: VSCode є кросплатформним середовищем розробки, яке можна встановити на різних операційних системах, включаючи Windows, MacOS та Linux.

Відкритість: VSCode є відкритим програмним забезпеченням, що означає, що його код доступний для перегляду та модифікації. Це дозволяє розробникам програмного забезпечення вносити свій внесок у розвиток середовища розробки та створювати власні додатки та розширення.

VS Code є корисним та потужним інструментом для розробки програмного забезпечення, включаючи мобільні відеоігри. BS Code має багато функцій та додатків, які можуть бути корисними для розробки мобільних відеоігор, таких як підтримка мови програмування Python, відлагодження, контроль версій, та інтеграція з іншими інструментами розробки.

2.3 Програмний модуль Pygame

Pygame - це не просто ще одна бібліотека для розробки ігор; це справжній феномен у світі програмування, що вже майже два десятиліття стоїть на сторожі мрій тисяч початківців розробників. Народившись на зорі нового тисячоліття, коли інтернет тільки починав свою експансію в домівки по всьому світу, Pygame став символом нової ери - ери, в якій створення відеоігор перестало бути прерогативою лише великих студій з багатомільйонними бюджетами.

Історія Pygame почалася в 2000 році, коли молодий американський програміст Піт Шинн вирішив, що світу Python не вистачає гідного інструменту для створення ігор. На той момент вже існували потужні фреймворки на C++ та Java, але вони часто були складними для новачків. Python же, зі своїм чистим синтаксисом та філософією "читабельності", стрімко набирав популярність серед початківців програмістів. Шинн бачив у цьому можливість: якщо дати цим новачкам правильні інструменти, вони зможуть не просто вчитися кодувати, але й втілювати свої ігрові фантазії в життя. Так на світ з'явився Pygame - спочатку як скромний проект, побудований на основі бібліотеки SDL (Simple DirectMedia Layer). SDL вже була відомою у світі C-програмування як надійний інструмент для роботи з мультимедіа, але її низькорівневий інтерфейс був не найдружнішим для початківців. Pygame взяв найкраще від SDL - її швидкість та надійність - і огорнув це в елегантний "пітонівський" інтерфейс.

Реліз Pygame 1.0 у квітні 2001 року не викликав фурору в індустрії, яка тоді була зосереджена на таких титанах, як "Grand Theft Auto III" або "Halo: Combat Evolved". Але в університетських кампусах та онлайн-форумах для початківців

програмістів Pygame швидко став культовим. Студенти, які раніше боролися з покажчиками в C++ або заплутувалися в абстрактних класах Java, раптом виявили, що можуть створити простий клон "Space Invaders" за один вікенд. Це було одкровенням.

Що ж робить Pygame таким особливим? По-перше, його філософія. Подібно до того, як Python прагне бути мовою, код якої читається "як англійська", Pygame прагне зробити розробку ігор настільки ж інтуїтивною. Візьмемо, наприклад, завдання відображення спрайту на екрані. У багатьох інших фреймворках це може вимагати розуміння буферів кадрів, текстурних координат та інших низькорівневих концепцій. У Pygame ж це просто: один рядок, і ваш герой магічно з'являється там, де ви хочете. Це не просто зручність; це свого роду емпауермент. Новачок бачить миттєвий результат своїх дій, що підсилює його впевненість і мотивацію продовжувати. Іншою ключовою перевагою Pygame є його модульність. Замість того, щоб нав'язувати розробнику певний підхід до створення гри, Pygame надає набір інструментів, які можна комбінувати на власний розсуд. Хочете створити стратегію реального часу? Використовуйте модулі Pygame для рендерингу тайлових карт і обробки введення миші. Мрієте про платформер? Комбінуйте спрайти, колізії та управління з клавіатури. Ця гнучкість означає, що Pygame не обмежує творчий потенціал розробника, а навпаки, підлаштовується під його бачення. Pygame також чудово демонструє принцип "легко вчитися, важко освоїти досконало". Візьмемо звук - критично важливий аспект будь-якої гри. Відтворити звуковий ефект у Pygame також просто: двох рядків достатньо, щоб ваш космічний корабель вибухнув з відповідним звуковим супроводом. Але під цією простотою ховається глибина. Хочете, щоб вибух звучав тихіше, якщо він далеко від гравця? Додайте панорамування та зміну гучності. Потрібні складні аудіоефекти для босів? Експериментуйте з частотною модуляцією та реверберацією. Pygame дозволяє початківцям швидко досягати результатів, але також залишає простір для росту.

З часом Pygame обріс багатьма потужними функціями. Його модуль “*draw*” дозволяє малювати примітиви - лінії, кола, багатокутники - що робить його чудовим інструментом для створення процедурно генерованих рівнів або ефектів частинок. Модуль “*transform*” полегшує масштабування, обертання та відображення спрайтів - необхідні операції для будь-якої динамічної гри. А “*mask*” дозволяє виконувати точне виявлення зіткнень на основі пікселів, що критично для платформерів або shoot 'em ups, де кожен піксель має значення.

Але, мабуть, найбільшим досягненням Pygame є не його технічні характеристики, а спільнота, яку він виростив. На відміну від багатьох інших галузей програмування, де новачки часто стикаються з елітарністю або токсичністю, спільнота Pygame славиться своєю доброзичливістю. На форумах Pygame, Reddit і Discord можна побачити, як досвідчені розробники терпляче пояснюють концепції колізій або оптимізації рендерингу тим, хто тільки починає свій шлях. Ця культура наставництва не лише допомагає людям вдосконалювати свої навички, але й надихає їх, у свою чергу, допомагати іншим.

Спільнота також робить величезний внесок у розширення можливостей Pygame. Офіційна бібліотека, хоч і потужна, має свої обмеження. Але завдяки відкритому характеру проекту, користувачі створили цілу екосистему додаткових модулів. Є “*pygame_gui*” для тих, хто хоче швидко створювати меню та інтерфейси. “*pygame-ce*” (Community Edition) додає підтримку сучасних функцій OpenGL. А “*pygame_sdl2*” дозволяє використовувати переваги останньої версії SDL, не чекаючи офіційних оновлень.

Особливо вражає те, як Pygame адаптувався до змін у ландшафті розробки ігор. Коли у 2010-х роках почався бум інді-ігор, багато розробників звернулися до Pygame для створення прототипів. Його швидкий цикл ітерацій ідеально підходив для методології "fail fast, iterate quickly". Гра не спрацьовує? Не страшно - з Pygame можна переробити її за вихідні.

З появою Raspberry Pi, крихітного, доступного комп'ютера, спрямованого на навчання дітей програмуванню, Pygame знайшов нову аудиторію. Його

невибагливість до ресурсів означає, що навіть на скромному обладнанні Рі діти могли створювати справжні ігри. Для багатьох це стало першим досвідом, коли їхній код оживав у формі, яка була більш захоплюючою, ніж просто текст у терміналі.

У середині 2010-х, коли "pixel art" відчув ренесанс в інді-сцені, Pygame знову виявився у потрібному місці. Його простий спрайтовий рушій ідеально підходив для ігор у стилі ретро. Такі хіти, як "Papers, Please" (хоча сама гра не була створена на Pygame), надихнули багатьох ентузіастів спробувати свої сили у створенні ігор з навмисно обмеженою естетикою. І для багатьох Pygame став воротами у цей світ.

Але, мабуть, найбільш несподіваним поворотом стала адаптація Pygame до ери машинного навчання. Коли такі бібліотеки, як TensorFlow і PyTorch, зробили глибоке навчання доступним для Python-розробників, дослідники ШІ виявили, що їм потрібне середовище для тестування своїх алгоритмів. І ось тут Pygame засяяв. Його простий, але гнучкий ігровий рушій ідеально підходив для створення середовищ, де агенти ШІ могли вчитися грати у прості ігри. Наприклад, дослідники використовували Pygame для створення клонів класичних ігор Atari, щоб навчити ШІ грати у них методом спроб і помилок.

Навіть у 2024 році, коли потужні ігрові рушії, такі як Unity та Unreal Engine, домінують у комерційній розробці ігор, Pygame все ще знаходить свою нішу. Для багатьох студентів комп'ютерних наук він слугує першим знайомством з основними концептами ігрової розробки - циклами подій, спрайтами, колізіями - без необхідності занурюватися у складнощі сучасних 3D-конвеєрів. А для досвідчених розробників Pygame залишається чудовим інструментом для швидкого прототипування та джемів.

Звичайно, у Pygame є свої обмеження. Його рендеринг, заснований на програмному забезпеченні, може боротися з великою кількістю спрайтів. Відсутність вбудованого редактора рівнів або інструментів анімації означає, що

розробникам часто доводиться створювати власні. А підтримка 3D, хоча і можлива через OpenGL, не є його сильною стороною.

Але, мабуть, в цьому і полягає чарівність Pygame. У світі, де ігрові рушії стають все більш складними, намагаючись реалізувати кожну можливу функцію, Pygame залишається вірним своїй філософії - надавати розробникам основні інструменти і дозволяти їм будувати на цій основі. Це стимулює творчість та винахідливість. Якщо вам потрібен редактор рівнів? Створіть його. Хочете складну систему частинок? Розробіть її самі. Це не просто навчає технічним навичкам; це виховує дух інженерного мислення.

У нашу еру, коли штучний інтелект обіцяє автоматизувати багато аспектів розробки програмного забезпечення, можна було б подумати, що такі інструменти, як Pygame, стануть застарілими. Але, можливо, все навпаки. У міру того, як ШІ бере на себе рутинні завдання, навички розуміння основ, творчого вирішення проблем та створення інструментів з нуля стають ще більш цінними. І в цьому сенсі Pygame, зі своїм мінімалістичним, але потужним набором інструментів, готує нове покоління розробників до майбутнього, де адаптивність та глибоке розуміння є ключовими.

У підсумку, Pygame - це більше, ніж просто бібліотека для розробки ігор. Це освітній інструмент, який навчив ціле покоління програмістів основам ігрового дизайну. Це полігон для інновацій, де розробники можуть експериментувати з новими ідеями. І, можливо найголовніше, це спільнота, яка вітає новачків і підтримує їх на шляху до того, щоб стати творцями. У світі, де технології часто здаються недосяжними, Pygame нагадує нам, що з правильними інструментами та підтримкою, кожен може стати творцем інтерактивних світів.

2.4 Інструмент для Піксельної графіки PixilArt

Піксельна графіка - це форма цифрового мистецтва, яка вже багато десятиліть залишається невід'ємною частиною світу відеоігор. Зародившись в

епоху, коли обмежені технічні можливості змушували розробників максимально економити кожен байт пам'яті, піксельна графіка перетворилася з технічної необхідності на повноцінний художній напрямок, який сьогодні переживає справжній ренесанс.

У своїй основі піксельна графіка - це техніка, де зображення створюється шляхом маніпуляції окремими пікселями, найменшими елементами цифрового зображення. Кожен піксель - це крихітний квадрат, який має певний колір і положення на сітці. Разом ці пікселі формують більш складні форми та об'єкти. На відміну від векторної графіки, де фігури визначаються математичними рівняннями, або сучасної растрової графіки високої роздільної здатності, де деталі можуть бути майже фотореалістичними, піксельна графіка навмисно підкреслює свою цифрову природу, роблячи піксельну структуру чітко видимою.

Історично піксельна графіка була єдиним доступним варіантом для ранніх комп'ютерних ігор та систем. Класичні консолі, такі як Nintendo Entertainment System (NES) або Sega Master System, мали дуже обмежені графічні можливості. Розробники були обмежені невеликою кількістю кольорів (часто не більше 16 або 32) і низькою роздільною здатністю екрану. Наприклад, NES мав роздільну здатність всього 256x240 пікселів. У таких умовах художники-піксельники стали справжніми майстрами мініатюри, здатними передати характер, емоції та динаміку руху всього кількома квадратиками.

Ця епоха породила іконічні образи, які й донині залишаються впізнаваними: Маріо та Луїджі з "Super Mario Bros.", Лінк з "The Legend of Zelda", Мегамен з однойменної серії ігор. Незважаючи на технічні обмеження, ці персонажі були сповнені життя та індивідуальності. Фони в іграх того часу також були шедеврами піксельного мистецтва - від таємничих підземель до казкових лісів, кожен рівень мав свій неповторний настрій.

З розвитком технологій у 1990-х та 2000-х роках індустрія відеоігор поступово перейшла на 3D-графіку та більш реалістичні 2D-зображення. Здавалося, піксельна графіка стала пережитком минулого. Однак, на початку

2010-х стався несподіваний поворот - піксельна естетика почала повертатися, тепер уже як свідомий художній вибір.

Цей ренесанс піксельної графіки був викликаний кількома факторами. По-перше, зросло покоління геймерів, які виросли на класичних 8-бітних і 16-бітних іграх. Для них піксельна графіка була пов'язана з теплими спогадами дитинства. По-друге, незалежні розробники (інді-студії) шукали способи виділитися на фоні високобюджетних проектів. Піксельна графіка пропонувала унікальний, миттєво впізнаваний стиль при значно менших витратах. По-третє, піксельна естетика стала символом певного бунту проти надмірної реалістичності та полірованості ігор-блокбастерів.

Такі ігри, як "Minecraft" (2011), "FEZ" (2012), "Papers, Please" (2013), "Shovel Knight" (2014) і "Stardew Valley" (2016), показали, що піксельна графіка може бути не просто ностальгічним трюком, а потужним інструментом для створення захоплюючих світів та атмосфер. Наприклад, "Papers, Please" використовує грубу, майже монохромну піксельну графіку, щоб передати гнітючу атмосферу бюрократії в тоталітарній державі. А яскраві, деталізовані пейзажі "Stardew Valley" створюють відчуття затишного сільського життя.

У цьому відродженні піксельного мистецтва важливу роль відіграють онлайн-інструменти, які роблять цю техніку доступною для кожного. Одним з найпопулярніших і найулюбленіших серед початківців і професіоналів є Pixilart - безкоштовний онлайн-сервіс, який переосмислює концепцію створення піксельної графіки для сучасної ери.

Pixilart був заснований у 2013 році, якраз на хвилі відродження піксельного мистецтва. Його творці поставили собі за мету демократизувати цю техніку, зробивши її доступною не лише для професійних художників, але й для всіх, хто хоче спробувати свої сили в піксельному мистецтві. Цей підхід повністю відповідає духу раннього періоду відеоігор, коли піксельна графіка була не просто мистецтвом, але й формою спільної творчості - художники, програмісти та

дизайнери рівнів тісно співпрацювали, разом вирішуючи, як найкраще використати кожен піксель.

Інтерфейс Pixilart вражає своєю простотою та інтуїтивністю. Тут немає складних меню чи заплутаних налаштувань. Відкриваючи сайт, користувач одразу бачить чисте полотно, розділене на сітку пікселів. Базові інструменти - пензель, гумка, заливка - знаходяться під рукою. Цей мінімалістичний підхід не випадковий: він нагадує про обмежені можливості ранніх графічних редакторів, але при цьому не відлякує новачків.

Однією з ключових переваг Pixilart є його онлайн-природа. У часи класичних ігор художники часто працювали на спеціалізованому обладнанні або складних програмах, які потребували потужних комп'ютерів. Pixilart же доступний через будь-який веб-браузер, на будь-якому пристрої - від настільного комп'ютера до смартфона. Немає потреби в установці програм або покупці ліцензій. Це особливо важливо для інді-розробників та любителів, які часто працюють над проектами у вільний час, на різних пристроях.

Попри простоту, Pixilart пропонує вражаючий набір інструментів. Тут є все необхідне для створення статичних зображень: різні пензлі, інструменти виділення, шари, можливість налаштування палітри. Палітра - це ключовий елемент в піксельному мистецтві. У старих іграх обмежена кількість кольорів змушувала художників ретельно продумувати кожен відтінок. Pixilart дозволяє працювати як з готовими палітрами, що імітують різні історичні системи (наприклад, EGA або палітру Game Boy), так і створювати власні.

Але справжня сила Pixilart розкривається при роботі з анімацією - невід'ємною частиною ігрової графіки. У класичних іграх кожен персонаж і об'єкт складався з набору спрайтів - невеликих зображень, які показувалися послідовно, створюючи ілюзію руху. Pixilart має вбудовані інструменти для покадрового створення анімації. Користувач може легко додавати нові кадри, копіювати та модифікувати існуючі, налаштовувати швидкість відтворення. Це дозволяє без

зусиль створювати цикли ходьби персонажів, анімації атак, ефекти вибухів - все те, що робить гру живою.

Соціальний аспект Pixilart теж має пряме відношення до світу відеоігор. Сервіс функціонує не просто як інструмент, а як спільнота. Користувачі можуть публікувати свої роботи, отримувати відгуки, брати участь у конкурсах. Багато художників-самоучок почали свій шлях саме тут, поступово удосконалюючи навички. Колаборація теж процвітає: одні створюють персонажів, інші - фони, треті - анімацію. Це нагадує про дух ранніх ігрових студій, де кожен вносив свій унікальний внесок у спільний проект.

Спосіб розробки ігор змінився: сьогодні багато інді-команд розподілені по всьому світу, спілкуючись переважно онлайн. Pixilart ідеально вписується в цю модель. Художник може створити концепт-арт персонажа в Pixilart, поділитися посиланням зі своєю командою, отримати відгуки, внести зміни - і все це в режимі реального часу, без необхідності встановлювати спільне програмне забезпечення.

Pixilart особливо цінний для розробників мобільних ігор, зокрема в таких жанрах як платформери або шутери. У світі, де більшість часу люди проводять зі смартфонами, піксельна графіка переживає друге народження. Вона чудово виглядає навіть на маленьких екранах, легко масштабується і не втрачає чіткості. Більше того, піксельні ігри, як правило, менш вимогливі до ресурсів, що дозволяє їм працювати навіть на бюджетних пристроях.

Але найголовніше - це настрій. У світі, де багато мобільних ігор виглядають однаково, з їх полірованою 3D-графікою та стандартними інтерфейсами, гра з піксельною графікою одразу виділяється. Вона викликає теплі спогади у старших гравців і пропонує щось свіже та унікальне молодшому поколінню. Це може бути платформер у стилі ретро, де герой долає перешкоди одним пікселем за раз, або динамічний шутер, де кожен піксельний постріл і вибух викликає хвилю ностальгії.

Варто зазначити, що Pixilart - не єдиний гравець на цьому полі. Існують й інші онлайн-інструменти, такі як Piskel, Aseprite (який, хоч і не є безкоштовним,

дуже популярний у професійних колах) або спеціалізовані програми на кшталт GraphicsGale. Але Pixilart виділяється своєю доступністю, фокусом на спільноту та чудовою підтримкою анімації.

Кожен рік на ринку з'являються десятки успішних ігор з піксельною графікою. У 2022 році такі проекти, як "Vampire Survivors" та "Dwarf Fortress" (Steam-версія), підкорили чарти продажів. І це не дивно. У світі, де технології дозволяють створювати майже фотореалістичні ігри, піксельна графіка пропонує щось інше - простір для уяви гравця.

Коли ми дивимося на піксельний пейзаж, наш мозок автоматично "заповнює" деталі, робить зображення більш живим, ніж воно є насправді. Це той самий ефект, який ми відчуваємо, читаючи книгу: слова дають нам основу, але саме наша уява робить історію унікальною. У цьому сенсі піксельна графіка - це майже літературний підхід до візуального мистецтва, і саме тому вона продовжує зачаровувати нових гравців.

Технічні обмеження, які колись змусили художників звернутися до піксельного мистецтва, тепер стали його найбільшою силою. У світі, де майже все можна візуалізувати з приголомшливою точністю, піксельна графіка нагадує нам про важливість творчої інтерпретації, про силу натяку та недомовленості. І такі інструменти, як Pixilart, гарантують, що це унікальне мистецтво буде процвітати й надалі, привносячи частинку цифрової магії в кожен піксель на наших екранах.

Рис. 2.1 Приклад спрайтів створених за допомогою PixilArt

3. ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Концепція гри

Концептуальна модель - це абстрактне представлення реального світу, системи або процесу, яке допомагає розуміти та аналізувати їхні характеристики, взаємозв'язки та особливості. Концептуальна модель є основою для розробки програмного забезпечення, баз даних, систем управління та інших комп'ютерних систем, оскільки вона дозволяє розробникам програмного забезпечення та іншим спеціалістам розуміти, як система працює та як вона може бути покращена.

Концептуальна модель складається з набору концептів, які представляють основні поняття та об'єкти системи, та взаємозв'язків між цими концептами, які представляють взаємодію та залежності між об'єктами системи. Концепти можуть бути представлені у вигляді діаграм, таблиць, текстових записів або інших форм, які допомагають розуміти та аналізувати систему.

Концептуальна модель є важливим інструментом для розробки програмного забезпечення, оскільки вона дозволяє розробникам програмного забезпечення розуміти, як система працює, як вона може бути покращена, та як вона може бути реалізована у вигляді програмного забезпечення. Концептуальна модель також допомагає розробникам програмного забезпечення спілкуватися з іншими спеціалістами, такими як бізнес-аналітики, менеджери проектів, та користувачі, оскільки вона забезпечує спільну мову та розуміння системи.

Так як дипломний проект реалізується одним студентом, діаграма має лише основні характеристики додатку.

Таблиця 3.1

Концептуальна таблиця гри

Назва гри	**Nolum`s mission**
жанр	platformer shooter
простір	2D
візуальний стиль	піксель арт
платформи	Android, Windows
мова програмування	Python
Технічні вимоги Технічні вимоги	Android 5.0 2 Гб оперативної пам’яті 100 Мб вільного простору
Ігровий процес	Гравець з’являється на карті з обмеженою кількістю боєприпасів, щоб закінчити гру потрібно перемогти всіх ворогів та дійти до фінішу
Сюжет	Гравець бере на себе роль солдата, який виконує секретне завдання в напівзруйнованому місті, що кишить лиходіями, його завдання знищити ворога та виконати завдання допоки місто вщент не буде зруйнованим

3.2 Функціонал Додатку

Розроблений продукт являє собою мобільний додаток у вигляді гри, що поєднує жанри платформера та шутера.

Основна мета самої гри – забезпечити гравцю цікавий та захоплюючий ігровий процес, а також надати можливість створювати та редагувати власні рівні.

Отже основні вимоги до продукту є такими:

- Різноманітність ігрових механік, таких як стрибки, стрільба, збір предметів, та інші елементи, щоб забезпечити цікавий та захоплюючий ігровий процес.
- Система прогресу, щоб заохочувати гравця досягати кращих результатів та продовжувати гру.
- Можливість створювати та редагувати власні рівні, щоб забезпечити безмежну реіграбельність та надати можливість гравцям проявляти свою творчість.
- Зрозуміле керування, щоб гравці могли легко та швидко освоювати гру.
- Якісна графіка, щоб забезпечити повне занурення у світ гри.
- Оптимізація та покращення продуктивності, щоб забезпечити плавний та стабільний ігровий процес на мобільних пристроях.

3.3 Діаграма use case

Use case діаграми - це графічні зображення, які показують, як користувачі взаємодіють з системою або програмним забезпеченням, щоб виконати певні завдання або дії. Use case діаграми є частиною об'єктно-орієнтованого підходу до розробки програмного забезпечення та допомагають розробникам програмного забезпечення розуміти, які функції та можливості потрібні для задоволення потреб та вподобань користувачів.

Use case діаграми складаються з наступних елементів:

Актори: це користувачі або інші системи, які взаємодіють з програмним забезпеченням. Актори можуть бути представлені у вигляді стікера або іконки.

Use case: це дії або завдання, які виконує актор, щоб взаємодіяти з програмним забезпеченням. Use case можуть бути представлені у вигляді овалу.

Відношення: це лінії, які з'єднують акторів та use case, щоб показати, як вони взаємодіють. Відношення можуть бути представлені у вигляді стрілки або лінії.

Use case діаграми можуть бути корисними для розробки програмного забезпечення, включаючи мобільні ігри в жанрі платформер шутер, оскільки вони допомагають розробникам програмного забезпечення розуміти, які функції та можливості потрібні для задоволення потреб та вподобань користувачів. Use case діаграми можуть бути використані для визначення функціональних вимог до програмного забезпечення, планування розробки, та тестування програмного забезпечення.

Для створення діаграми було визначено одного актора – користувача, який буде грати в гру. Інших акторів не передбачено, оскільки гра не підтримує багатокористувацький режим і не використовує бази даних.

Щоб створити діаграму, також потрібно описати всі можливі варіанти використання системи:

Варіанти використання в головному меню:

- Перехід до головного меню
- Перехід до меню створення рівня
- Перехід до меню редагування рівня
- Перехід до меню початку гри
- Перехід до меню налаштувань

Варіанти використання в меню створення рівня:

- Створення нового рівня
- Варіанти використання в меню редагування рівня:
- Редагування існуючого рівня
- Перехід до меню початку гри

Варіанти використання в меню початку гри:

- Запуск гри
- Перехід на новий рівень

Варіанти використання в грі:

- Перехід до кінця гри

Варіанти використання в меню кінця гри:

- Перехід до головного меню

Варіанти використання в меню налаштувань:

- Перехід до головного меню

На основі сформованих вхідних даних і списку варіантів використання можна створити діаграму використання, на якій ми побачимо, як саме актор може взаємодіяти з грою.

Опис діаграми

Діаграма демонструє різні сценарії використання гри. Основний актор – Гравець – взаємодіє з наступними функціональними можливостями:

Головне меню:

Гравець входить в головне меню, звідки може перейти до створення рівня, редагування рівня, початку гри або налаштувань.

Створення рівня:

Гравець може створити новий рівень. Після цього він може перейти до редагування рівня або початку гри.

Редагування рівня:

Гравець може редагувати існуючий рівень. Після редагування він може перейти до початку гри.

Початок гри:

Гравець може розпочати гру або перейти на новий рівень.

Кінець гри:

Після завершення гри гравець може повернутися до головного меню.

Налаштування:

Гравець може налаштувати параметри гри і повернутися до головного меню.

Ця діаграма відображає логіку взаємодії гравця з грою, забезпечуючи послідовність дій та можливості для користувача.

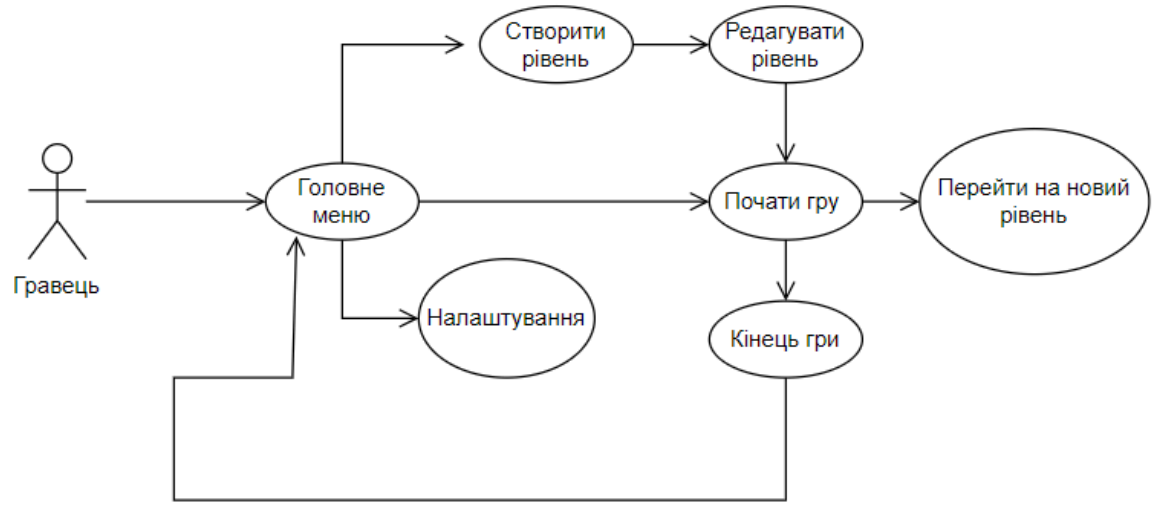


Рис. 3.1 use case діаграма

3.4 Діаграма активності

Діаграма, що представлена на даному малюнку, пояснює, як саме користувач взаємодіє з програмою для створення рівнів у грі, починаючи з запуску програми і до основного циклу гри.

Після запуску програми, вона починає з налаштування екрану і меж, завантаження зображень, створення кнопок і ініціалізації змінних. Після цього програма переходить до головного циклу.

У головному циклі відбуваються наступні дії:

- Малювання фону.
- Малювання сітки.
- Малювання світу.
- Малювання тексту інтерфейсу.

Після малювання інтерфейсу, програма перевіряє, чи було натискання кнопок. Залежно від того, яку кнопку було натиснуто, виконуються різні дії:

Якщо натиснути кнопку гри, то виконується функція play game.

При натисканні кнопки збереження, зберігаються дані світу у форматі CSV.

Якщо ж натиснути кнопку завантаження, завантажуються дані світу.

Після перевірки кнопок, програма оновлює екран і перевіряє події. Якщо гравець вирішив перейти до гри, то здійснюється перехід у гру, інакше головний цикл продовжується.

Таким чином, ця діаграма активності ілюструє основні етапи роботи програми: від запуску до головного циклу, малювання інтерфейсу, обробки натискань кнопок і можливості переходу безпосередньо до гри.

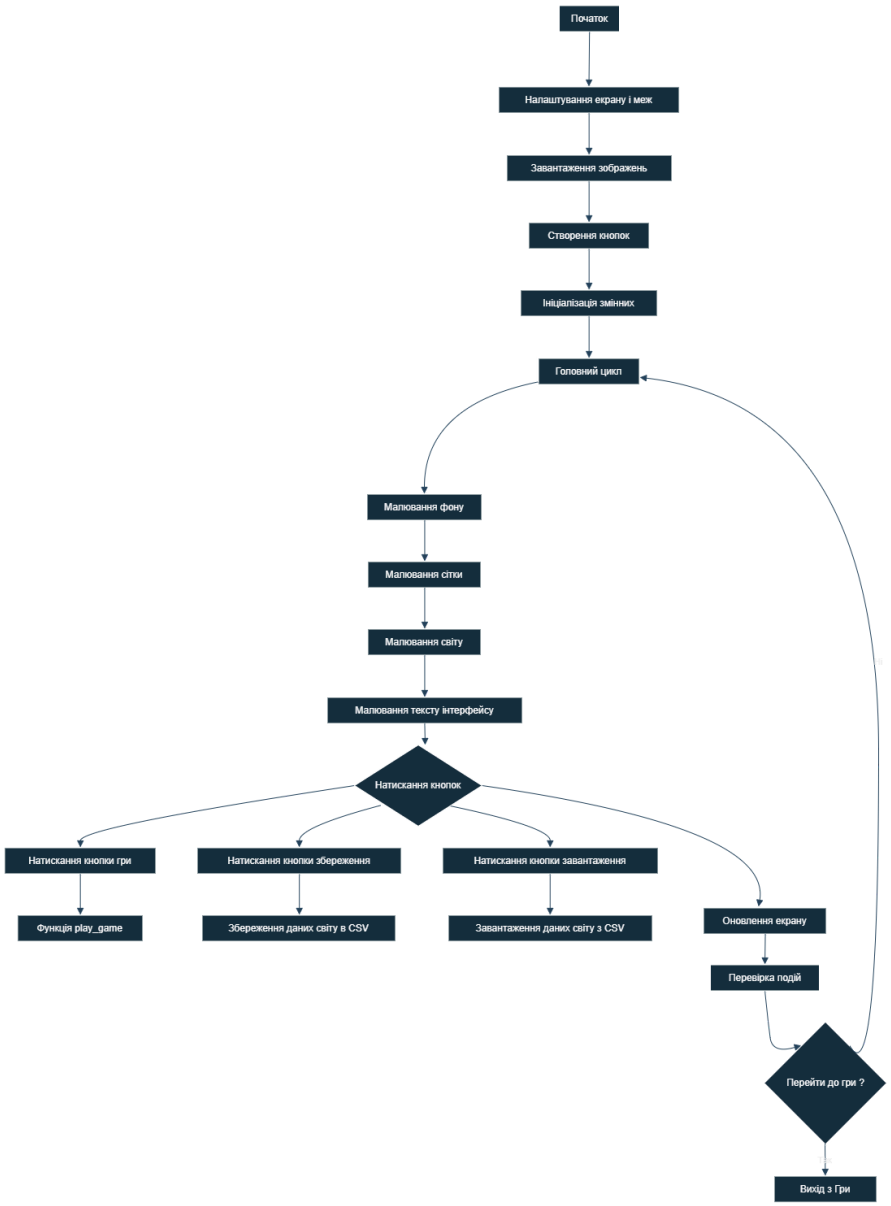


Рис. 3.2 Діаграма активності редактора рівнів

3.5 діаграма класів

Діаграма класів на малюнку ілюструє основні класи, що використовуються для створення гри, та їх взаємозв'язки. Розглянемо детальніше кожен клас та їх ролі.

Soldier - Це ключовий клас, який представляє солдата в грі. Клас містить інформацію про стан (alive, health), характеристики (speed, ammo, grenades), а також методи для оновлення, руху, стрільби, анімації та перевірки стану живого.

Enemy - Клас, що представляє ворогів у грі. Містить подібні до Soldier атрибути і методи, такі як health, speed, ammo, direction, та методи для руху, стрільби та оновлення анімації.

Bullet - Клас, який представляє кулю в грі. Має атрибути для зберігання швидкості кулі, напрямку і зображення, а також метод для оновлення положення кулі.

Grenade - Клас, що представляє гранату в грі. Містить таймер, швидкість, напрямок і метод для оновлення положення гранати.

Explosion - Клас для обробки вибухів. Зберігає зображення вибуху, індекс кадру анімації та метод для оновлення анімації вибуху.

HPBar - Клас, що відповідає за відображення шкали здоров'я. Містить атрибути для позиції, здоров'я та метод для малювання шкали здоров'я на екрані.

Box - Клас, що представляє коробки з предметами. Має атрибути для типу предмета, зображення, положення та методи для ініціалізації та оновлення стану коробки.

Decorate - Клас для декоративних елементів у грі. Містить зображення та метод ініціалізації.

Exit - Клас для позначення виходу на рівні. Має зображення та метод ініціалізації.

World - Клас, що представляє світ гри. Зберігає список об'єктів (obs list) і містить методи для обробки даних світу та їх малювання.

Усі ці класи взаємодіють між собою, щоб створити повноцінну гру. Наприклад, клас Soldier використовує Bullet для стрільби, а також може взаємодіяти з класами HPBar для відображення стану здоров'я. Вороги (Enemy) мають подібну функціональність до солдатів, що дозволяє їм стріляти і переміщатися. Клас World управляє всіма об'єктами на рівні, координуючи їх поведінку і взаємодію. Класи Decorate та Exit додають елементи оформлення і функціональні об'єкти на рівні.

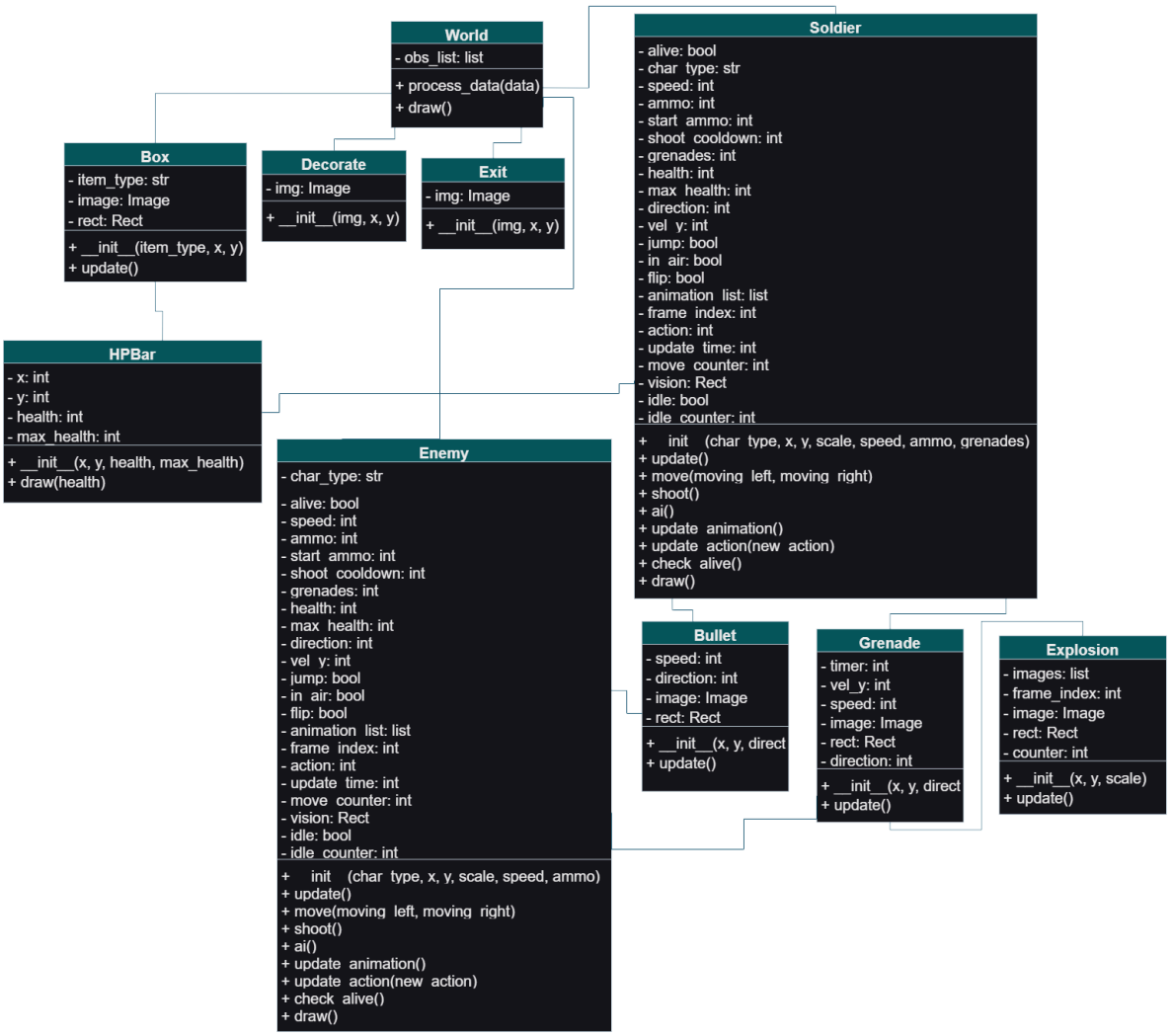


Рис. 3.3 Діаграма класів нащадків

4. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Підготовка розробки додатку

Підготовка до реалізації проекту досить клопіткий процес планування, організації, та підготовки необхідних ресурсів та інструментів для розробки та реалізації програмного забезпечення або відеоігри. Підготовка до реалізації проекту є важливим кроком для успішної розробки та реалізації проекту, оскільки вона допомагає розробникам програмного забезпечення та дизайнерам відеоігор розуміти, що потрібно зробити, як це зробити, та що потрібно для цього.

Підготовка до реалізації проекту може включати такі кроки:

- Визначення цілей та завдань проекту: розробники програмного забезпечення та дизайнери відеоігор повинні визначити цілі та завдання проекту, включаючи функціональність, дизайн, терміни, та бюджет.
- Планування проекту: розробники програмного забезпечення та дизайнери відеоігор повинні розробити план проекту, який включає етапи, терміни, ресурси, та бюджет.
- Вибір інструментів та технологій: розробники програмного забезпечення та дизайнери відеоігор повинні вибрати необхідні інструменти та технології для розробки та реалізації проекту, включаючи мову програмування, середовище розробки, бібліотеки, фреймворки, та інші інструменти.
- Підготовка ресурсів: розробники програмного забезпечення та дизайнери відеоігор повинні підготувати необхідні ресурси для розробки та реалізації проекту, включаючи людські ресурси, обладнання, програмне забезпечення, та інші ресурси.
- Організація співпраці: розробники програмного забезпечення та дизайнери відеоігор повинні організувати співпрацю між членами команди,

включаючи розподіл обов'язків, комунікацію, та спільне використання ресурсів.

GitHub - це безкоштовна веб-платформа для спільної розробки програмного забезпечення, яка дозволяє розробникам програмного забезпечення та дизайнерам відеоігор спільно працювати над проектами, ділитися кодом, відстежувати зміни, та співпрацювати над розробкою програмного забезпечення або відеоігри. GitHub є корисним інструментом для підготовки до реалізації проекту, оскільки він дозволяє розробникам програмного забезпечення та дизайнерам відеоігор.

Спільна робота над проектом: GitHub дозволяє розробникам програмного забезпечення та дизайнерам відеоігор спільно працювати над проектом, ділитися кодом, та співпрацювати над розробкою програмного забезпечення або відеоігри. А також дає доступ до бібліотек, фреймворків, та інших інструментів, щоб ефективніше працювати над проектом.

Відстеження змін: GitHub дозволяє розробникам програмного забезпечення та дизайнерам відеоігор відстежувати зміни в коді, дізнатися, хто зробив зміни, та чому були зроблені зміни.

Співпраця: GitHub дозволяє розробникам програмного забезпечення та дизайнерам відеоігор співпрацювати над розробкою програмного забезпечення або відеоігри, ділитися ідеями, та надавати зворотний зв'язок.

Організування репозиторії: GitHub дозволяє розробникам програмного забезпечення та дизайнерам відеоігор організувати репозиторії, щоб легко знайти та відстежувати код, документацію, та інші ресурси.

Керування версіями: GitHub використовує систему контролю версій Git, яка дозволяє розробникам створювати різні гілки (branches) проекту, експериментувати з новими функціями, не впливаючи на основний код, а потім об'єднувати (merge) ці гілки, коли функція готова.

Управління проектами: GitHub пропонує інструменти для управління проектами, такі як дошки завдань (project boards), мітки (labels), та віхи

(milestones), які допомагають командам організовувати і відстежувати прогрес роботи.

Обговорення та рецензування коду: Функції GitHub, такі як Pull Requests та Code Reviews, дозволяють розробникам обговорювати запропоновані зміни, рецензувати код один одного та забезпечувати його якість перед інтеграцією в основний проект.

Автоматизація процесів: GitHub Actions дозволяє автоматизувати робочі процеси розробки, такі як тестування, збірка та розгортання, що значно економить час і зусилля.

Документування: GitHub надає можливість створювати та розміщувати документацію безпосередньо в репозиторії, використовуючи GitHub Pages та README файли, що робить інформацію про проект легкодоступною.

Співпраця з відкритим кодом: GitHub є центром open-source проектів, де розробники можуть знайти тисячі проектів з відкритим кодом, до яких вони можуть долучитися, або використовувати їх як основу для власних проектів.

Навчання та розвиток: GitHub пропонує ресурси для навчання, такі як GitHub Learning Lab, де розробники можуть вдосконалювати свої навички програмування та співпраці через практичні завдання.

Безпека та контроль доступу: GitHub надає різні рівні доступу для репозиторіїв, двофакторну аутентифікацію, та інтеграцію з інструментами безпеки коду, щоб захистити проекти та особисті дані.

Інтеграція з іншими інструментами: GitHub інтегрується з багатьма популярними інструментами розробки, такими як Jira, Slack, та Azure DevOps, що дозволяє створити єдиний робочий процес.

Аналітика та інсайти: GitHub надає аналітичні інструменти, які показують внески кожного члена команди, популярність репозиторію, та іншу корисну статистику для оцінки здоров'я проекту.

GitHub - це не просто платформа для зберігання коду; це повноцінна екосистема для розробки програмного забезпечення, яка сприяє співпраці,

прозорості та ефективності. Вона є незамінним інструментом для розробників та дизайнерів, які працюють над складними проектами, такими як відеоігри, де координація та контроль версій є критично важливими.

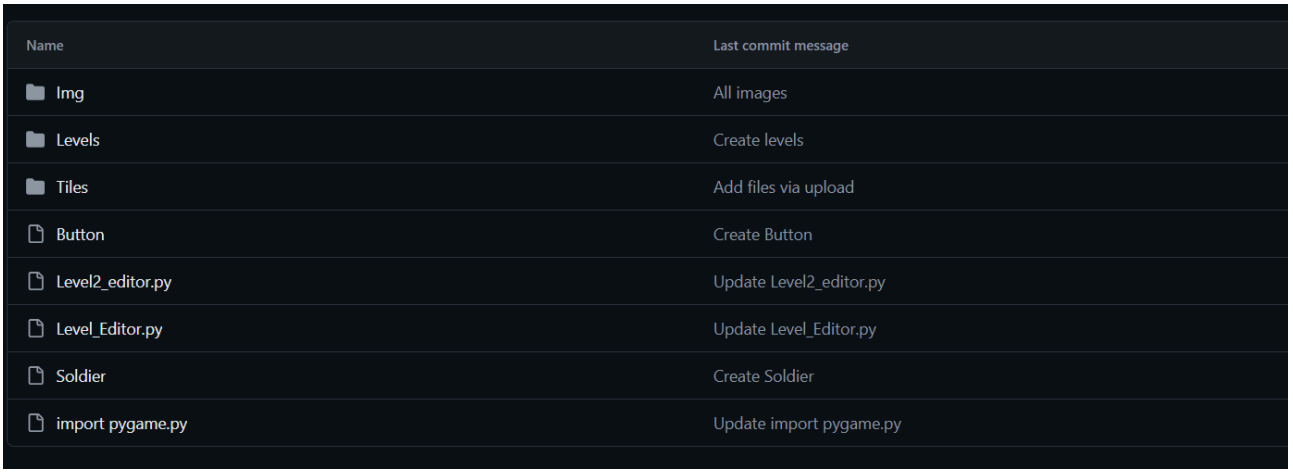


Рис. 4.1 приклад використання GitHub для керування версіями додатку

4.2 Написання Коду

Написання коду є ключовим елементом в розробці будь-якої відеогри. Код є основою для всіх функцій, механік, та взаємодій в грі, та забезпечує її працездатність та стабільність.

Для прикладу можна навести декілька основних елементів для яких потрібен код:

Реалізація функцій та механік: код є необхідним для реалізації всіх функцій та механік в грі, включаючи рух персонажа, стрільбу, фізику, анімацію, та інші елементи.

Взаємодія з користувачем: код є необхідним для реалізації взаємодії з користувачем в грі, включаючи керування персонажем та редагування рівнів

Інтеграція з іншими інструментами та технологіями: код є необхідним для інтеграції з іншими інструментами та технологіями, які використовуються в грі, особливо графіку.

Оптимізація та покращення продуктивності: код є необхідним для оптимізації та покращення продуктивності гри, включаючи зменшення часу завантаження, збільшення кількості кадрів в секунду, та зменшення споживання пам'яті.

Написання коду в додатку повинно бути ретельно продуманим та організованим, щоб забезпечити його працездатність та стабільність. Важливо використовувати відповідні інструменти та технології, такі як мова програмування, середовище розробки, бібліотеки, фреймворки, та інші інструменти, щоб ефективніше працювати над проектом. Також важливо дотримуватися найкращих практик написання коду, таких як коментування та форматування, щоб забезпечити чистоту, зрозумілість, та надійність коду.

4.2.1 Клас кнопки

За створення інтерактивних кнопок на екрані гри відповідає клас Button. Цей клас дозволяє відображати кнопки та обробляти їхні натискання користувачем. Він забезпечує створення інтерактивних кнопок, які реагують на натискання миші, що дозволяє реалізувати різноманітні функції в грі, наприклад, перехід між екранами або виконання певних дій при натисканні на кнопку.

```python
import pygame

class Button():
    def __init__(self,x, y, image, scale):
        width = image.get_width()
        height = image.get_height()
        self.image = pygame.transform.scale(image, (int(width * scale), int(height * scale)))
        self.rect = self.image.get_rect()
        self.rect.topleft = (x, y)
        self.clicked = False

    def draw(self, surface):
        action = False


        pos = pygame.mouse.get_pos()

        if self.rect.collidepoint(pos):
            if pygame.mouse.get_pressed()[0] == 1 and not self.clicked:
                action = True
                self.clicked = True

        if pygame.mouse.get_pressed()[0] == 0:
            self.clicked = False


        surface.blit(self.image, (self.rect.x, self.rect.y))

        return action
```

Рис. 4.2 Реалізація класу Button

4.2.2 Клас Солдат

Клас Soldier використовується для створення об'єктів, що представляють головного героя та ворогів у грі. Ці об'єкти мають різноманітні властивості та поведінку, включаючи базову штучну інтелекту (AI) для ворогів. Клас Soldier забезпечує базову функціональність для створення персонажів у грі, включаючи рух, стрибки, атаки, анімації та просту поведінку AI для ворогів.

```python
class Soldier(pygame.sprite.Sprite):
    def __init__(self, char_type, x, y, scale, speed, ammo, grenades):
        pygame.sprite.Sprite.__init__(self)
        self.alive = True
        self.char_type = char_type
        self.speed = speed
        self.ammo = ammo
        self.start_ammo = ammo
        self.shoot_cooldown = 0
        self.grenades = grenades
        self.health = 100
        self.max_health = self.health
        self.direction = 1
        self.vel_y = 0
        self.jump = False
        self.in_air = True
        self.flip = False
        self.animation_list = []
        self.frame_index = 0
        self.action = 0
        self.update_time = pygame.time.get_ticks()
        #for ai
        self.move_counter = 0
        self.vision = pygame.Rect(0, 0, 150, 20)
        self.idle = False
        self.idle_counter = 0
```

Рис. 4.3 Реалізація класу Soldier

4.2.3 Функція для Штучного Інтелекту ворогів

Метод ai визначає поведінку ворожого персонажа (AI) в грі. Цей метод керує рішеннями ворога щодо руху та атаки на гравця. Він також відповідає за визначення діапазону видимості ворога. Цей метод визначає простий, але ефективний механізм руху та поведінки ворогів у грі, що дозволяє їм реагувати на дії гравця та оточуючий світ.

```python
def ai(self):
    if self.alive and player.alive:
        if not self.idle and random.randint(1, 100) == 1:
            self.update_action(0) #0=idle
            self.idle = True
            self.idle_counter = 50
        if self.vision.colliderect(player.rect):
            self.update_action(3) #3=attack
            self.shoot()
        else:
            if not self.idle:
                if self.direction == 1:
                    ai_move_right = True
                else:
                    ai_move_right = False
                ai_move_left = not ai_move_right
                self.move(ai_move_left, ai_move_right)
                self.update_action(1) #1=run
                self.move_counter += 1
                self.vision.center = (self.rect.centerx + 75 * self.direction, self.rect.centery)


                if self.move_counter > TILE_SIZE:
                    self.direction *= -1
                    self.move_counter *= -1
            else:
                self.idle_counter -= 1
                if self.idle_counter <= 0:
                    self.idle = False
```

Рис. 4.4 Реалізація класу Ai

4.2.4 Клас кулі

Клас Bullet використовується для створення об'єктів куль у грі. Ці об'єкти відповідають за рух та взаємодію з іншими об'єктами у грі, такими як гравець та вороги. Цей клас дозволяє створювати та керувати кулями у грі, що важливо для реалістичної бойової системи, де кулі можуть взаємодіяти з іншими об'єктами та наносити шкоду.

```python
class Bullet(pygame.sprite.Sprite):
    def __init__(self, x, y, direction):
        pygame.sprite.Sprite.__init__(self)
        self.speed = 10
        self.image = bullet_img
        self.rect = self.image.get_rect()
        self.rect.center = (x, y)
        self.direction = direction

    def update(self):
        self.rect.x += (self.direction * self.speed)
        if self.rect.right < 0 or self.rect.left > SCREEN_WIDTH:
            self.kill()

        for tile in world.obs_list:
            if tile[1].colliderect(self.rect):
                self.kill()

        if pygame.sprite.spritecollide(player, bullet_group, False):
            if player.alive:
                player.health -= 5
                self.kill()
        for enemy in enemy_group:
            if pygame.sprite.spritecollide(enemy, bullet_group, False):
                if enemy.alive:
                    enemy.health -= 25
                    self.kill()
```

Рис. 4.5 Реалізація класу Bullet

4.2.5 клас гранати

Клас Grenade відповідає за створення об'єктів гранат у грі. Ці об'єкти можуть рухатися та взаємодіяти з іншими об'єктами, наносячи шкоду гравцям та ворогам. Цей клас реалізує логіку руху та вибуху гранати, що додає динаміку та стратегічний аспект у грі, дозволяючи гравцям та ворогам уникати або використовувати гранати для стратегічних цілей.

```
class Grenade(pygame.sprite.Sprite):
    def __init__(self, x, y, direction):
        pygame.sprite.Sprite.__init__(self)
        self.timer = 100
        self.vel_y = -11
        self.speed = 15
        self.image = grenade_img
        self.rect = self.image.get_rect()
        self.rect.center = (x, y)
        self.width = self.image.get_width()
        self.height = self.image.get_height()
        self.direction = direction

    def update(self):
            self.vel_y += GRAVITY
            dx = self.direction * self.speed
            dy = self.vel_y

            for tile in world.obs_list:
                if tile[1].colliderect(self.rect.x + dx, self.rect.y, self.width, self.height):
                    self.direction *= -1
                    dx = self.direction * self.speed

                if tile[1].colliderect(self.rect.x, self.rect.y + dy, self.width, self.height):
                    self.speed = 0
                    if self.vel_y < 0:
                        self.vel_y = 0
                        dy = tile[1].bottom - self.rect.top
                    elif self.vel_y >= 0:
                        self.vel_y = 0
                        dy = tile[1].top - self.rect.bottom
```

Рис. 4.6 Реалізація класу Grenade

Продовження класу гранат

```
            self.rect.x += dx
            self.rect.y += dy

            self.timer -= 1
            if self.timer <= 0:
                self.kill()
                explosion = Explosion(self.rect.x, self.rect.y, 0.5)
                explosion_group.add(explosion)
                if abs(self.rect.centerx - player.rect.centerx) < TILE_SIZE * 2 and \
                    abs(self.rect.centery - player.rect.centery) < TILE_SIZE * 2:
                        player.health -= 50
                for enemy in enemy_group:
                    if abs(self.rect.centerx - enemy.rect.centerx) < TILE_SIZE * 2 and \
                        abs(self.rect.centery - enemy.rect.centery) < TILE_SIZE * 2:
                            enemy.health -= 50
```

Рис. 4.7 Реалізація класу Grenade

4.2.1 клас вибух

А також окремий клас Explosion, що відповідає за створення анімації вибуху у грі. Ці об'єкти відтворюють послідовність зображень, що створюють ефект вибуху. Цей клас дозволяє створювати реалістичний ефект вибуху у грі, що підсилює візуальний ефект та динаміку подій.

```
class Explosion(pygame.sprite.Sprite):
    def __init__(self, x, y, scale):
        pygame.sprite.Sprite.__init__(self)
        self.images = []
        for num in range(1, 6):
            img = pygame.image.load(f'img\explosion\exp{num}.png').convert_alpha()
            img = pygame.transform.scale(img, (int(img.get_width() * scale), int(img.get_height() * scale)))
            self.images.append(img)
        self.frame_index = 0
        self.image = self.images[self.frame_index]
        self.rect = self.image.get_rect()
        self.rect.center = (x, y)
        self.counter = 0

    def update(self):
        EXPLOSION_SPEED = 4
        self.counter += 1
        if self.counter >= EXPLOSION_SPEED:
            self.counter = 0
            self.frame_index += 1
            if self.frame_index >= len(self.images):
                self.kill()
            else:
                self.image = self.images[self.frame_index]
```

Рис. 4.8 Реалізація класу Explosion

ВИСНОВКИ

Після завершення роботи над дипломним проектом "Розробка мобільної гри в жанрі платформер шутер з можливістю створення та редагування власних рівнів", можна зробити такі висновки:

Результат роботи: в результаті роботи було розроблено повноцінну мобільну гру в жанрі платформер шутер з можливістю створення та редагування власних рівнів. Гра відповідає всім вимогам, вказаним в завданні, та має високу якість та стабільність роботи.

Для розробки гри були використані такі інструменти та технології: мова програмування Python, бібліотека Pygame, середовище розробки Visual Studio Code, система контролю версій Git, та платформа для спільної розробки Github.

Розроблена гра має різноманітні ігрові механіки, такі як стрибки, стрільба, збір предметів, та інші елементи, щоб забезпечити цікавий та захоплюючий ігровий процес.

Розроблений додаток має систему прогресу та нагород, щоб заохочувати гравця досягати кращих результатів та продовжувати гру. Проект має якісну графіку, щоб забезпечити повне занурення у світ гри.

Загалом, дипломний проект "Розробка мобільної гри в жанрі платформер шутер з можливістю створення та редагування власних рівнів" є успішним та перспективним. Розроблена гра має високу якість та стабільність роботи, а також відповідає всім вимогам, вказаним в завданні. Розробка таких ігор дозволяє не тільки задовольнити потреби та вподобання цільової аудиторії, але також сприяє розвитку навичок та творчості гравців.

Робота пройшла апробацію:

Комісарук А.В., Залива В. В. Використання штучного інтелекту для розробки ігор. IV Всеукраїнська науково-практична конференція «Сучасні інтелектуальні інформаційні технології в науці та освіті». 15 травня 2024 р., Київ,

Державний університет інформаційно-комунікаційних технологій. Збірник тез К.: ДУІКТ, 2024. С. 157-158.

Комісарук А.В., Залива В. В. Порівняння мов програмування для розробки мобільних ігор. IV Всеукраїнська науково-практична конференція «Сучасні інтелектуальні інформаційні технології в науці та освіті». 15 травня 2024 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез К.: ДУІКТ, 2024. С. 299-300.

ПЕРЕЛІК ПОСИЛАНЬ

1. Watkiss S. Beginning Game Programming with Pygame Zero. Berkeley, CA : Apress, 2020. URL: https://doi.org/10.1007/978-1-4842-5650-3 (date of access: 28.05.2024).
2. Gold M. Creating Video Games Using PyGame: A comprehensive guide to creating your own games in python / Mike Gold., 2023. – 219 c.
3. Parker J. Game Development Using Python / James Parker., 2021. – 338 c. – (2nd edition).
4. UML базові знання [Електронний ресурс] – Режим доступу до ресурсу: https://www.uml-diagrams.org/.
5. PixilArt [Електронний ресурс] – Режим доступу до ресурсу: https://www.pixilart.com/.
6. Free 2D-game assets [Електронний ресурс] – Режим доступу до ресурсу: https://craftpix.net/.
7. Side-scrolling video games wikipedia [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Side-scrolling_video_game.
8. Visual Studio Code documentation [Електронний ресурс] – Режим доступу до ресурсу: https://code.visualstudio.com/docs.
9. Python documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3/.
10. Pygame documentation [Електронний ресурс] – Режим доступу до ресурсу: https://www.pygame.org/docs/ref/pygame.html.
11. Kivy’s documentation [Електронний ресурс] – Режим доступу до ресурсу: https://kivy.org/doc/stable/.
12. GitHub documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.github.com/en.

13. Класифікація ігор за жанрами [Електронний ресурс] – Режим доступу до ресурсу: https://training.qatestlab.com/blog/technical-articles/games-genres/.

14. Video games terms and concepts [Електронний ресурс] – Режим доступу до ресурсу: https://www.encyclopedia.com/science-and-technology/technology/technology-terms-and-concepts/video-games.

15. Python Game Development Tutorials [Електронний ресурс] – Режим доступу до ресурсу: https://realpython.com/tutorials/gamedev/.

16. Python Game Development Libraries [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/python-game-development-libraries/.

17. Video game industry - Statistics & Facts [Електронний ресурс] – Режим доступу до ресурсу: https://www.statista.com/topics/868/video-games/#topicOverview.

18. Game Development with Python [Електронний ресурс] – Режим доступу до ресурсу: https://zeba.academy/python-game-development/.

19. Shinde P. N. Pygame: Develop Games using Python. International Journal for Research in Applied Science and Engineering Technology. 2021. Vol. 9, no. VI. P. 4442–4447. URL: https://doi.org/10.22214/ijraset.2021.35735.

20. Kelly S. Python, PyGame, and Raspberry Pi Game Development. Berkeley, CA : Apress, 2019. URL: https://doi.org/10.1007/978-1-4842-4533-0.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Створення мобільної гри жанру 2d scroll platform shooter за допомогою мови програмування Python

Виконав студент 4 курсу
групи ПД-41
Комісарук Антон Володимирович
Керівник роботи
Старший викладач кафедри ІПЗ Залива Віталій Вікторович

Київ – 2024

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи:** підвищення зацікавленості потенційних гравців можливістю створення власних рівнів та їх подальшого редагування.
- **Об'єкт дослідження:** геймплей, що базується на проходженні платформних рівнів з можливістю їх створення та редагування.
- **Предмет дослідження:** програмне забезпечення, що дозволяє створювати та редагувати рівні для самостійного проходження гри в жанрі platform shooter

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Провести аналіз існуючих ігор в жанрі платформер-шутер та визначити їх сильні та слабкі сторони.
2. Вивчити доступні технічні засоби для розробки ігор та вибрати найкращі для реалізації проекту.
3. Визначити основні параметри гри, включаючи жанр, простір, вид графіки та ігрові особливості.
4. Спроектувати гру, враховуючи обрані параметри та технічні засоби.
5. Реалізувати функціонал гри, включаючи створення та редагування рівнів.

Концепт гри

1. Гравець керує персонажем у жанрі платформер-шутер, що може стрибати, бігати і стріляти або кидати гранати.
2. Головна мета гравця — пройти рівні, знищуючи ворогів та долаючи перешкоди на шляху.
3. Гравець має можливість створювати і редагувати власні рівні за допомогою вбудованого редактора рівнів.
4. У процесі гри гравець може збирати різні предмети, які допомагають покращити здібності персонажа або отримати нову зброю.
5. Після завершення кожного рівня гравець може перейти до наступного рівня або повернутися до редактора для створення нових рівнів.
6. Гра продовжується, поки гравець не пройде рівень або не буде знищений.

АНАЛІЗ АНАЛОГІВ

Характеристика	**Strike Force Heroes**	**Contra**	**Nolum`s mission**
Платформа	Веб-браузер	ПК, ігрові консолі	ПК, Мобільні пристрої
Мова програмування	ActionScript 3.0	Ассемблер, C	Python
Кількість персонажів	1-5	1-2	1
Можливість кастомізації рівнів	–	–	+
Наявність бонусів	+	+	+
Індикатори характеристик	+	–	+
Реіграбельність	–	–	+

ВИМОГИ ДО ГРИ

1. Різноманітність ігрових механік, таких як стрибки, стрільба, збір предметів, та інші елементи, щоб забезпечити цікавий та захоплюючий ігровий процес.
2. Система прогресу, щоб заохочувати гравця досягати кращих результатів та продовжувати гру.
3. Можливість створювати та редагувати власні рівні, щоб забезпечити безмежну реіграбельність та надати можливість гравцям проявляти свою творчість.
4. Зрозуміле керування, щоб гравці могли легко та швидко освоювати гру.
5. Якісна графіка, щоб забезпечити повне занурення у світ гри.
6. Оптимізація та покращення продуктивності, щоб забезпечити плавний та стабільний ігровий процес на мобільних пристроях.

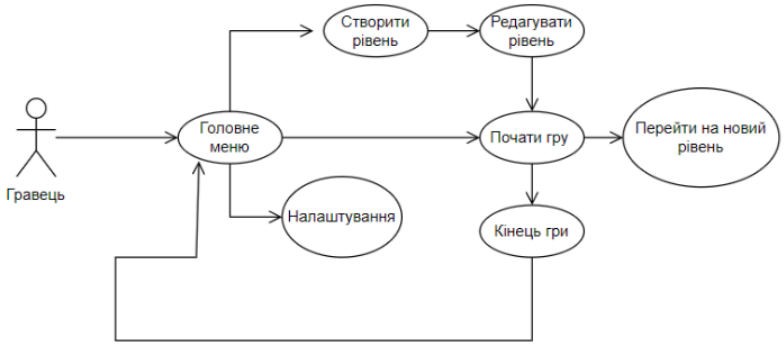
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

Visual Studio Code

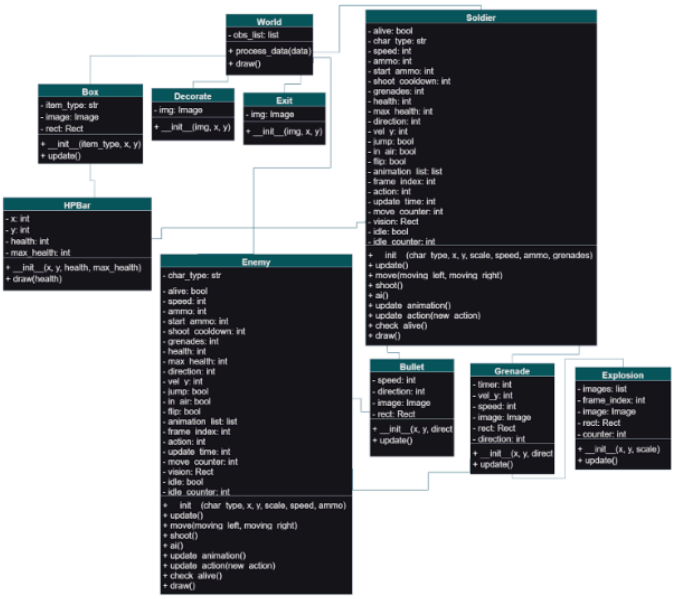
pygame

python

Діаграма використання



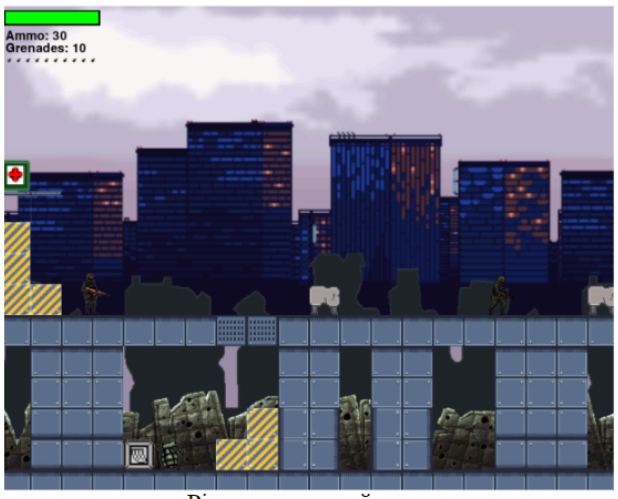
Діаграма класів



Екранні форми

Екран редагування рівнів

Екранні форми



Рівень створений гравцем

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Комісарук А.В. Використання штучного інтелекту для розробки ігор / Комісарук А.В. Залива В. В. // Розробка ігор та ігрофікація освітніх та бізнес процесів: Матеріали IV Всеукраїнській науково-практичній конференції «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез 15 травня 2024 року, ДУІКТ, м. Київ – К: ДУІКТ, 2024 С. 157-158.
2. Комісарук А.В. Порівняння мов програмування для розробки мобільних ігор / Комісарук А.В. Залива В. В. // Розробка ігор та ігрофікація освітніх та бізнес процесів: Матеріали IV Всеукраїнській науково-практичній конференції «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез 15 травня 2024 року, ДУІКТ, м. Київ – К: ДУІКТ, 2024. С. 299-300.

ВИСНОВКИ

1. Під час дослідження ринку мобільних ігор було виявлено, що жанр 2D platform shooter є популярним серед гравців, але все ще існує простір для нових ідей та підходів. Потенційна аудиторія гри складається з гравців віком від 16 до 35 років, які цікавляться шутерами та платформерами.
2. Аналіз існуючих мобільних ігор, що належать до жанру схожого до мого, дозволив визначити їхні переваги та недоліки. Наприклад, гра Strike Force Heroes має гарну графіку та різноманітні рівні, але її керування не завжди зручне для мобільних пристроїв. Гра Contra має просту та інтуїтивно зрозумілу механіку, але її графіка та звук застарілі. Ці висновки були взяті до уваги при розробці концепції та дизайну моєї гри.
3. Розробка концепції гри дозволила створити цікавий сюжет, персонажів з унікальними здібностями та ігрову механіку, яка поєднує в собі елементи шутерів та платформерів. Дизайн рівнів був розроблений таким чином, щоб вони були цікавими та викликали у гравців бажання пройти їх повністю.
4. Створення прототипу гри та тестування його на користувачах дозволило отримати цінний зворотний зв'язок та визначити недоліки, які потрібно було виправити. Наприклад, гравці відзначили, що керування потрібно було зробити більш зручним для мобільних пристроїв, а також що потрібно було додати більше бонусів та патронів.
5. Розробка дизайну гри дозволила створити анімацію графіку які доповнюють ігрову механіку та створюють цікаву атмосферу гри.
6. Тестування та відлагодження гри дозволило виправити помилки та поліпшити продуктивність. Гра працює на мобільних пристроях з різними характеристиками.

ДЯКУЮ ЗА УВАГУ!