МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
(повне найменування назва факультету)

Кафедра програмування
(повна назва кафедри)

ДИПЛОМНА РОБОТА

Створення гри за допомогою рушія Unity

Виконав: студент групи ПМІ-41
спеціальності 122 – комп'ютерні науки
(шифр і назва спеціальності)

Бурдяк О.В.
(підпис) (прізвище та ініціали)

Керівник Ярошко С.А.
(підпис) (прізвище та ініціали)

Рецензент
(підпис) (прізвище та ініціали)

2023

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Кафедра програмування

Спеціальність 122 Комп'ютерні науки

«ЗАТВЕРДЖУЮ»
Завідувач кафедри Ярошко С.А.

" " 2022 року

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Бурдяку Олегу Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Створення гри за допомогою рушія Unity»

керівник роботи Ярошко Сергій Адамович, доцент, к. ф.-м. н. ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені Вченою радою факультету від "13"_вересня 2022 року №_15_.

2. Строк подання студентом роботи 12 червня 2023 року

3. Вихідні дані до роботи літературні джерела, інтернет-ресурси, постановка задачі, наукові статті

4. Зміст дипломної роботи (перелік питань, які потрібно розробити)

1. Дослідити ігровий рушій Unity;

2. Програмно реалізувати гру та оптимізувати її для смартфонів;

3. Провести тестування застосунку.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Ознайомлення з предметною областю	05.09.2022 – 11.09.2022	виконано
2	Написання специфікації вимог	20.09.2022 – 22.09.2022	виконано
3	Вибір технологій	23.09.2022 – 26.09.2022	виконано
4	Дослідження ігрового движка	01.10.2022 – 25.10.2022	виконано
5	Написання сценарію гри	06.01.2023 – 11.01.2023	виконано
6	Програмна реалізація застосунку	12.01.2023 – 25.03.2023	виконано
7	Тестування отриманих результатів	25.03.2023 – 05.04.2023	виконано
8	Оформлення дипломної роботи	16.05.2023 – 07.06.2023	виконано
9	Подання дипломної роботи	12.06.2023	виконано

Студент _____ Бурдяк О.В.

(підпис) (прізвище та ініціали)

Керівник роботи _____ Ярошко С.А.

(підпис) (прізвище та ініціали)

ЗМІСТ

АВТОРЕФЕРАТ

Ця дипломна робота присвячена розробці 2D мобільної гри за допомогою ігрового рушія Unity.

Перший розділ досліджує основи створення ігор і специфіку рушія Unity. Включає детальний аналіз основних складових частин Unity: його інтерфейсу, методів створення сцен та об'єктів. Особлива увага приділяється програмуванню в Unity за допомогою мови програмування C#, що дозволяє керувати поведінкою ігрових об'єктів, взаємодією з користувачем та логікою гри.

У другому розділі описується процес дизайну та реалізації гри. Представлено концепт гри, включаючи рівні, персонажів, об'єкти, звуки та інші елементи, що складають цілісну та занурювальну гру. Подробиці програмної реалізації, використання ресурсів, оптимізація для платформи Android, та процес збірки описані в цьому розділі.

Третій розділ присвячений тестуванню гри, його мета – переконатися в стабільності, продуктивності та коректності реалізації геймплею. Додатково, цей розділ включає інструкцію з використання гри, призначену для кінцевих користувачів.

ВСТУП

Оцінка сучасного стану об'єкта розробки. Ні для кого не секрет що комп'ютерні ігри отримали величезну популярність у XXI столітті. У них грають дорослі й діти, у вільний і робочий час. Дехто грає щоб відпочити, дехто щоб розважитись, а для декого це спосіб заробітку. Що ж до мінусів комп'ютерних ігор це те, що в користувача погіршується стан його здоров'я, втрачається продуктивність праці, та й не завжди є доступ до ПК. Тому із появою смартфонів все дедалі більшої популярності набирають мобільні ігри, оскільки користуватися мобільним пристроєм можна де завгодно.

Актуальність роботи та підстави для її виконання. В умовах цьогорічної війни ігри відіграють чималу роль в підтриманні психічного здоров'я багатьох людей. При довгому перебуванні в укритті чи при переїзді до іншого міста з'являється можливість виникнення стресу у дітей. Саме ігри допомагають їм у знятті стресу, оскільки граючи вони опрацьовують свої емоції і запобігають виникненню різних факторів стресу.

У сучасному світі величезна кількість дітей має доступ до ігор на мобільних пристроях. За іграми які містять розвивальний контент приховується величезна користь. Граючи в такі ігри діти, а також дорослі розвивають особисту творчість та деякі когнітивні здібності. Можна виділити, що особливо розвивається візуально - просторова здатність, яка сприяє запам'ятовуванню об'єктів і зв'язків між ними.

Мета й завдання роботи. Метою цієї роботи являється створення 2D мобільної гри за допомогою ігрового рушія Unity.

Для досягнення поставленої мети необхідно виконати такі завдання:

- Дослідити методи створення ігор.
- Детально вивчити особливості роботи з ігровим рушієм Unity.
- Дослідити усі методи тестування ігор.
- Розробити користувацький інтерфейс та наглядно продемонструвати його.
- Програмно реалізувати застосунок.
- Оптимізувати застосунок для коректної роботи на смартфоні.

- Створити збірку гри на смартфон з операційною системою Android.
- Провести тестування отриманих результатів.

РОЗДІЛ 1. ОЗНАЙОМЛЕННЯ ІЗ СЕРЕДОВИЩЕМ РОЗРОБКИ UNITY ТА ДОСЛІДЖЕННЯ МЕТОДИК ІГРОВОГО ТЕСТУВАННЯ

1.1. Що таке Unity

Перед початком написання гри потрібно визначити які існують види розробки застосунку. Для виконання поставленої задачі виділяють 2 способи її реалізації, таких як написати гру з нуля або використовувати рушій. Що ж на рахунок першого способу - це дуже затратно по часу, відповідно погіршується якість роботи, тому було прийнято рішення реалізувати поставлену задачу за допомогою ігрового рушія. Далі потрібно визначити який саме рушій обрати. Технологію слід вибирати, опираючись на вимоги нашого застосунку. Для реалізації нашої задачі ідеально підійде ігровий рушій Unity.

Так що ж таке Unity? Unity - це професійний багатоплатформний інструмент для розробки ігор. Тобто у цьому рушію можна створювати застосунки для персональних комп’ютерів, мобільних пристроїв, ігрових консолей. Він особливий тим, що його програмне забезпечення є дуже потужним, простим і одночасно безплатним для використання до тих пір, коли ви не почнете заробляти великі гроші з його допомогою. В основному суть його експлуатації закладається в розробці 2d/3d ігор.

Можна виділити, що об’єкт містить власну потужну кросплатформну IDE для розробки, це означає те, що в самому середовищі є багато важливих вбудованих функцій. Неймовірно багато часу займає створення якоїсь звичайної фізики або ж, наприклад, 3-D рендерингу, саме в цьому відображається вся користь цієї IDE. В Unity є величезна бібліотека ассетів “Asset Store”, де ж самі автори мають можливість ділитись готовими скриптами, модельками, матеріалами та всім своїм творінням. Звичайно, що вони мають також змогу отримати все готове з цього магазину, купивши товар, або за безплатно.

1.2. Unity, інтерфейс середовища

Після встановлення Unity постає необхідність встановити одну із запропонованих його версій, а після цього створити новий проєкт.

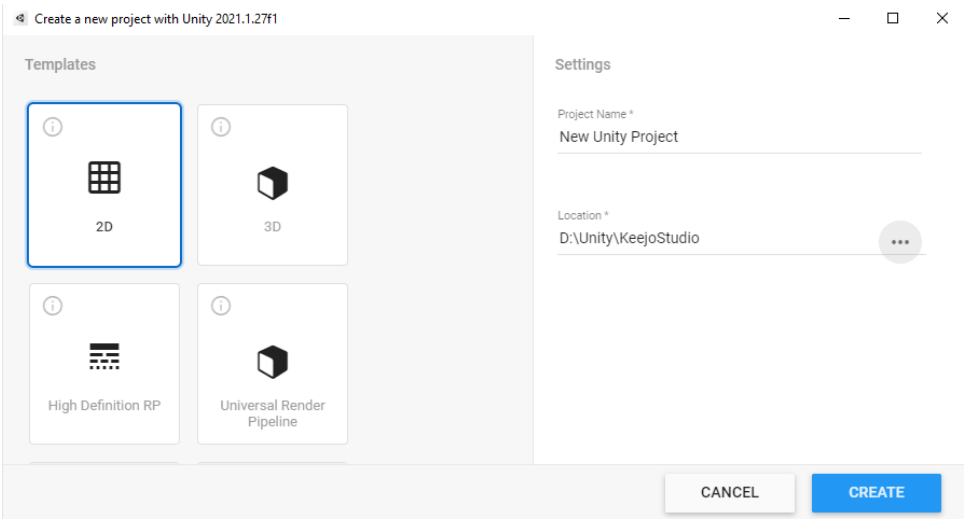

Рис. 1.1. Вікно створення порожнього застосунку для рушія Unity

Спершу нам потрібно вибрати шаблон гри, придумати назву проєкту (не самого застосунку) і вибрати папку, у якій ми будемо зберігати нашу гру (рис. 1.1). Далі, відкривши створений проєкт в Unity, ми зустрінемо декілька основних вікон, піктограм і параметрів (рис. 1.2).

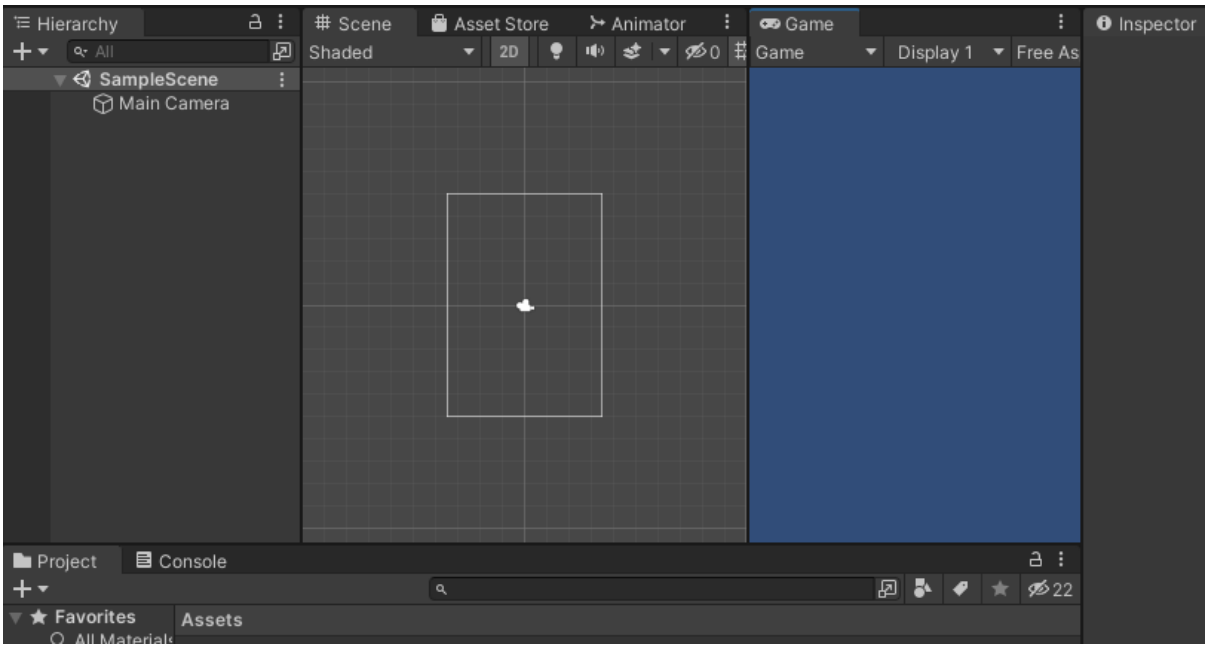

Рис. 1.2. Вікно рушія Unity

- Hierarchy: за замовчуванням тут відображається довгий список усіх наших ігрових об’єктів у сцені. Завдяки цьому списку легше знайти будь-який наш об’єкт, щоб змінити його властивості, або ж додати компоненти.
- Scene: це найбільше вікно, яке розміщене посередині, у ньому ми можемо спостерігати поточний рівень, або меню. Також у цьому вікні ми можемо вільно транспортувати, збільшувати та зменшувати наші об’єкти з Ієрархії.
- Game: у цьому вікні ми будемо мати таку ж перспективу, як камера, і не зможемо пересувати предмети.
- Asset Store: це вікно із вищезгаданим магазином ассетів для розробників.
- Inspector: після того, як ми вибираємо об’єкт з ієрархії, саме у вікні інспектор ми можемо змінювати властивості цього об’єкту. Тобто додавання скриптів, зміна розміру, транспортування і додавання готових компонентів.
- Projeckt: це вікно розташоване внизу, й у ньому зображаються усі наявні сцени, звуки, папки, картинки, анімації тощо. Саме у цьому вікні створюються нові сценарії C#.
- Console: тут показуватимуться помилки, попередження, або ж інші повідомлення від редактора.

1.3. Створення сцен та об’єктів в Unity

Процес створення нової сцени в Unity вимагає переходу в меню “File”, де користувач повинен вибрати “New Scene”. Сцена є основою для розробки ігор в Unity, яка може включати різні об'єкти, такі як герої, вороги, декорації, освітлення, звуки та інші елементи, необхідні для створення ігрового середовища.

Додавання об'єктів до сцени є критичним аспектом розробки ігор у цьому рушії. Об'єкти можуть бути створені вручну, імпортовані з зовнішніх джерел або ж взяті з вбудованого магазину “Asset store”. Для створення нового об'єкта необхідно перейти в меню “GameObject” і вибрати потрібний тип об'єкта.

Об'єкти в Unity мають різні властивості, які можна налаштувати за допомогою інспектора. Властивості можуть включати параметри, такі як позицію, розмір, колір, текстуру, матеріал, фізичні властивості та інші, що можуть бути визначені розробником.

Успішна розробка гри вимагає не тільки створення і розташування об'єктів, але й контролювання їхньої поведінки. Для цього необхідно писати скрипти, які визначають, як об'єкти взаємодіють один з одним та з ігровим середовищем.

Щоб гравець міг спостерігати за сценою, в Unity використовуються камери. Позиція камери, її орієнтація та інші параметри визначають, яку частину сцени бачить гравець.

1.4 Програмування в Unity за допомогою C#

Однією з фундаментальних частин розробки в Unity є програмування. Unity використовує мову програмування C# для створення скриптів, що регулюють поведінку об'єктів в ігровому середовищі.

C# є об'єктно-орієнтованою мовою програмування, яка є популярною в індустрії розробки ігор завдяки своїй гнучкості, потужності та зрозумілості. Ця мова надає розробникам можливість писати високоякісні скрипти для взаємодії з різними компонентами гри, забезпечуючи таким чином потрібну поведінку.

Процес програмування в Unity починається зі створення нового скрипту через меню “Assets”, “Create” і далі “C# Script”. Новостворений скрипт можна приєднати до об'єкта в ігровій сцені, перетягнувши його на об'єкт у вікні Ієрархії, або ж через Інспектор.

Всередині скрипта визначаються методи, які є діями, виконуваними об'єктом. Зокрема, поширені методи, як “Start” та “Update”, використовуються для виконання коду на етапах початку гри та під час кожного кадру відповідно.

Метод “Start” використовується для ініціалізації змінних, налаштування початкового стану об'єкта, або для виконання будь-якого коду, що повинен відбутися на початку гри.

“Update”, з іншого боку, використовується для опису коду, який виконується у кожному кадрі, і відіграє ключову роль в управлінні поведінкою об'єктів в реальному часі. Це може включати обробку вводу користувача, рух об'єктів, взаємодію між об'єктами та багато іншого.

Скрипти в Unity можуть взаємодіяти з багатьма аспектами ігрового двигуна, включаючи об'єкти, компоненти, фізику, вхідні дані та аудіо, що надає розробникам глибокий контроль над ігровим досвідом.

1.5 Методика тестування ігор

Через різноманітну варіативність ігрових подій часто трапляються помилки або ж різні дефекти. Для виявлення та усунення цих проблем існує необхідність у тестуванні.

Розрізняють декілька ролей серед тестувальників ігор:

- Game QA - це тестувальник, до обов’язків якого належить виявлення та документація в bug - report різних дефектів. Він відповідає за те, щоб гра не крашилась, а дії персонажів мали сенс.
- Lead tester - несе велику відповідальність, тісно співпрацює із художниками й гейм-дизайнерами. Дивиться за тим, щоб не існувало дублікатів із bug - report.

- Game Producer - встановлює терміни на виявлення та виправлення помилок. Встановлює терміни релізу застосунку.

Також виділяють кілька основних видів тестувань:

- Kombinatorial Testing - можна назвати методом експериментального проєктування. Відповідає за створення систематичних комбінацій для створення тестів.
- Functional Testing - виконання цього методу тестування є необхідним, а також на цей метод виділяють чимало часу, оскільки тут тестувальники шукають проблеми в самій грі, зв’язані з ігровим процесом, графікою та аудіо. Перевіряється, чи поставлена задача спрацювала відповідно до специфікацій сценарію.
- Compatibility Testing - цей метод тестування використовується задля перевірки працездатності додатка на певних пристроях. Тестувальники перевіряють, чи користувацький інтерфейс відповідає програмним розмірам екрана пристрою. Перевіряють видимість і доступність всього контенту застосунку.
- Load Testing - метод, необхідний здебільшого для тестувань онлайн гри. Цю технологію тестування використовують, щоб визначити продуктивність додатка в режимі реального часу. Перевіряють стійкість системи під час максимального навантаження користувачами.
- Clean Room Testing - це один із небагатьох методів тестування, де ж саме тестування відбувається без участі тестувальника. Тобто створюються тести, які будуть грати в гру так, як грають користувачі.

РОЗДІЛ 2. РОЗРОБКА ЗАСТОСУНКУ

2.1. Опис додатка

Додаток містить сцену із користувацьким інтерфейсом, а також 6 ігрових сцен. Він призначений виключно для мобільних пристроїв на операційній системі android версії від 5.5. Гра належить до категорії музичних ігор, її цільова аудиторія для дітей віком від 3 років.

Суть гри полягає в тому, щоб вчасно попадати по випадаючим різноколірним стрілкам згори екрану та заробляти ігрові очки й монети. Гра завершується у тому випадку, коли перестає грати музика. При завершенні гри появляється віконце із кількістю зароблених очків і монет. Також передбачено, що при гральному процесі на складніших рівнях, реалізовано передчасне завершення гри при певній кількості промахів по стрілках. В такому випадку появляється анімована панелька з надписом “You Lose” і з кнопками перегратиі й виходом у головне меню. При вдалих завершеннях гри ми заробляємо монети та маємо можливість їх потратити на розблокування наступних рівнів, або ж розблокування нових персонажів у ігровому магазині.

Складність гри збільшується відносно рівня, а також при виборі складності до рівня. Відповідно цьому пришвидшується випадання стрілок і збільшується їхня кількість. Коли ж вибрана складність “Hard”, тоді повзунок слайдера, який розміщений вгорі, стає “більш чутливим”, і вже при невеликій кількості промахів можливо програти.

2.2. Реалізація користувацького інтерфейсу

Жоден додаток не може існувати без свого користувацького інтерфейсу. Він не повинен бути складним, а, навпаки, легким і зрозумілим. Отже, інтерфейс нашого додатка складається із 2 основних вікон, таких як: головне меню і магазин. В головному меню у нас є панель з монетами, а також значок магазину. Також у нас

для кожного рівня є можливість обрати його одну із 3 видів складності “Easy”, “Normal”, “Hard”. Також на початку гри доступний лише перший рівень, решту потрібно купити за монети.

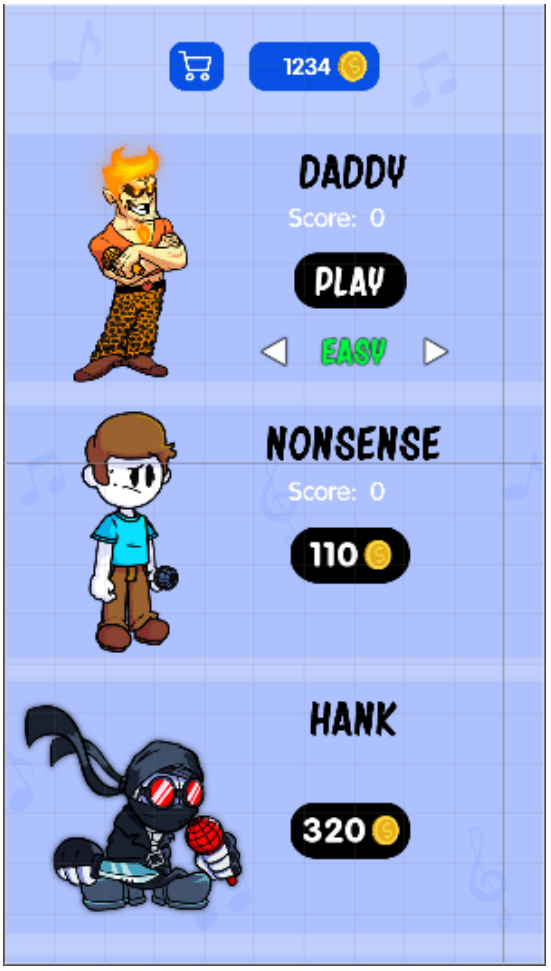

Рис. 2.1. Користувацький інтерфейс, головне меню

Оскільки виникає проблема в розміщенні усіх ігрових рівнів на головному меню, було добавлено об’єкт з компонентом “ScrollRect”, який дозволить нам скролити вміст самого об’єкту. Для нього було створено дочірній об’єкт “Content” з компонентом “Vertical Layout Group”, який розміщує свої дочірні елементи макета один на одному.

Рис. 2.2. Користувацький інтерфейс, вікно магазину

Якщо ми перейдемо у вікно магазину, ми бачимо, що по центрі є панель із головним персонажем, під ним можуть бути кнопки такі як: “Buy” або ж “Set”, в залежності чи персонаж куплений, чи ні. Також для зручності навігації по боках розміщені кнопки для переходу між різними ігровими персонажами.

2.3. Програмна реалізація основних механік застосунку

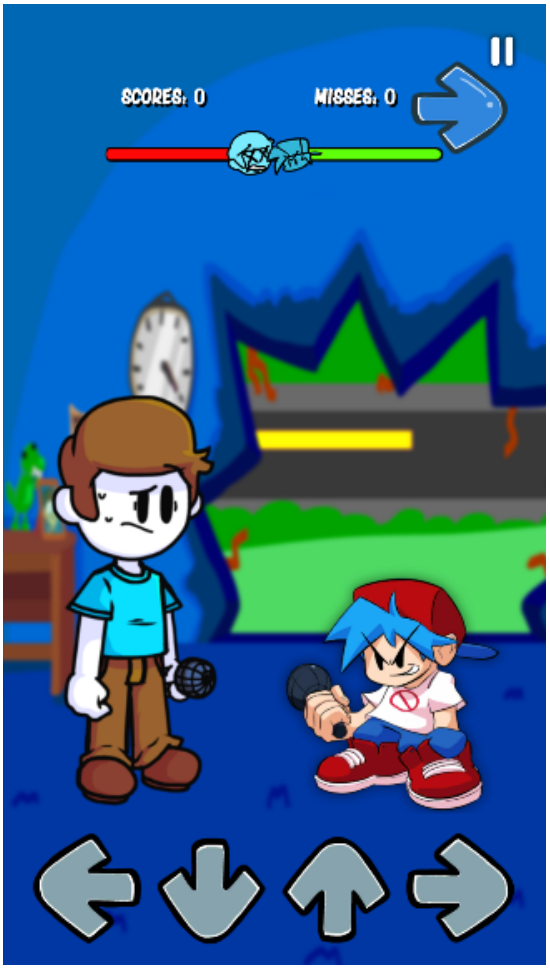

Рис. 2.3. Вигляд грального процесу на другому рівні

Для того, щоб додаток працював відповідно заданим умовам, потрібно його програмно реалізувати. На ігрових сценах зображені два об’єкти, це головний герой і супротивник, для обох них були реалізовані анімації. У вікні “Animator” ми розробили декілька анімацій для кожного з них (рис. 2.4). У вікні “Parameters” створено 5 тригерів для доступності анімацій в коді.

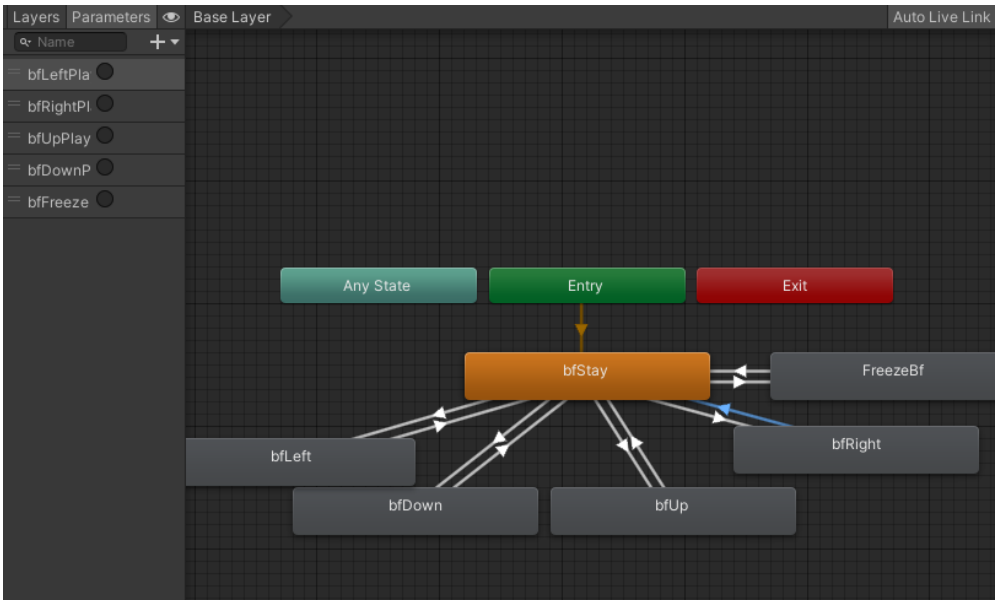

Рис. 2.4. Вікно з реалізацією анімацій для головного персонажа

Для реалізації випадання стрілок було створено декілька десятків ігрових об’єктів, кожен з яких мав свою фізику за допомогою добавлення в об’єкт RidgetBody. Також були доданими колайдери для відстеження взаємодії стрілок з іншими об’єктами в сцені. Коли стрілка падає до самого низу і доторкається до невидимого об’єкта “DestroyersArrow”, в якого також є колайдер, а тег змінений на “Destr”, реалізовано видалення стрілки. При взаємодії колайдера стрілки з колайдером об’єкта “DestroyersArrow”, виконується перевірка по тегу об’єкта. Якщо тег об’єкта з яким взаємодії стрілки являється “Destr”, то знищуємо саму стрілку (рис. 2.5).

```
71      private void OnTriggerEnter2D(Collider2D collision)
72      {
73          if (collision.tag == "Destr")
74          {
75              Destroy(gameObject);
76              myOver.AddMissCounter();
77              mechaniksSlider.controlSlider();
78              myOver.comboes = 0;
79          }
80          if (collision.tag == "AddPoint")
81          {
82              PlayBF((int)collision.gameObject.layer);
83              Destroy(gameObject);
84              myOver.AddPointCounter();
85              myOver.comboes += 1;
86              myBollForCheck = false;
87              myBollForCheck1 = false;
88          }
```

Рис. 2.5. Програмна реалізація взаємодії колайдера стрілок із іншими колайдерами

Оскільки застосунок тісно зв’язаний з музикою, у ньому було реалізовано декілька класів, які відповідають за музику, а також звукові ефекти.

Вгорі по центрі розміщений слайдер, по якому користувач може орієнтуватись в прогресі попадань по стрілках (рис. 2.3). Для випадку, коли суб’єкт погано гратиме і не буде попадати по стрілках вчасно, реалізовано, що повзунок слайдера буде переміщатись до правого кута, а якщо дійде до кінця, то гра буде завершеною.

У розробленому додатку була реалізована система покупок. Ця система була розроблена за допомогою декількох власних класів, у тому числі класу PlayerPrefs, вбудованого в Unity, що використовується для зберігання даних на пристроях користувачів.

Одним з ключових компонентів цієї системи є клас YouWin. Цей клас відповідає за обробку подій виграшу в рівні, включаючи підрахунок та зберігання зароблених монет. В контексті цього класу, реалізовано два методи: RestarLevel і BackToMenu. Вони активуються відповідно при рестарті рівня та поверненні до головного меню.

Спочатку визначається загальна кількість монет користувача, яка є сумою поточної кількості монет та кількості монет, зароблених за поточний рівень. Потім

ця сума зберігається в системі PlayerPrefs для подальшого використання. На завершення, в залежності від того, який метод був активований, гра або перезапускає поточний рівень, або повертається до головного меню (рис. 2.6).

```
80      public void RestarLevel()
81      {
82          myMoney = PlayerPrefs.GetInt("money") + currentMyMoney;
83          PlayerPrefs.SetInt("money", myMoney);
84          SceneManager.LoadScene(SceneManager.GetActiveScene().name);
85      }
86
        0 references
87      public void BackToMenu()
88      {
89          myMoney = PlayerPrefs.GetInt("money") + currentMyMoney;
90          PlayerPrefs.SetInt("money", myMoney);
91          SceneManager.LoadScene(0);
92      }
```

Рис. 2.6. Код що демонструє процес обробки та збереження даних

2.4 Оптимізація

Для того щоб гра працювала безперебійно на різних смартфонах нам потрібно її оптимізувати. Проаналізуємо стан нашої гри. Запустивши нашу гру в Unity, натискаємо на кнопку Stats щоб отримати дані про стан гри (рис. 2.7).

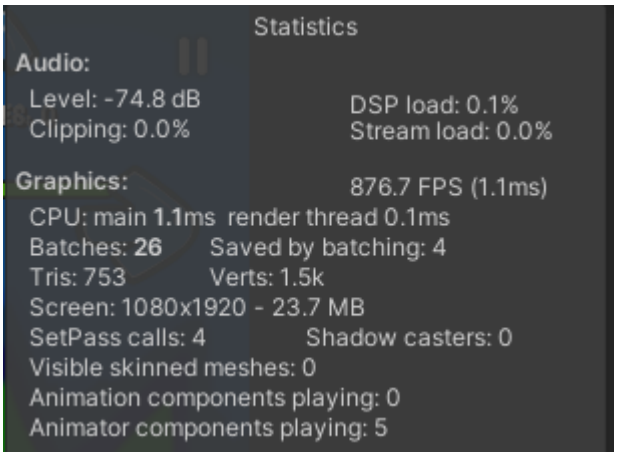

Рис. 2.7. Статистична інформація про стан гри в Unity з використанням вбудованого інструменту “Stats”

Таблиця 3.1

Порівняння дійсних характеристик ігри та їх допустимих норм

Характеристики	Отримані значення	Допустима норма
Batches	26	< 200
FPS	876.8	> 30
Verts (кількість верши)	1.5k	< 100k
SetPass calls	3	< Batches

З таблиці 3.1 видно, що всі отримані показники відповідають встановленим нормам для мобільних ігор на платформі Android. Кількість SetPass calls значно менша за кількість Batches, що також відповідає вимогам ефективності.

Далі переходимо до оптимізації користувацького інтерфейсу. Одним з основних аспектів, на який слід звернути увагу, є адаптивність інтерфейсу для різних розмірів екрана. Метою є гарантувати, що всі елементи інтерфейсу відображаються належним чином та зберігають свою пропорційність на різних пристроях. Крім того, це охоплює забезпечення того, що елементи інтерфейсу не накладаються один на одного незалежно від розміру екрана.

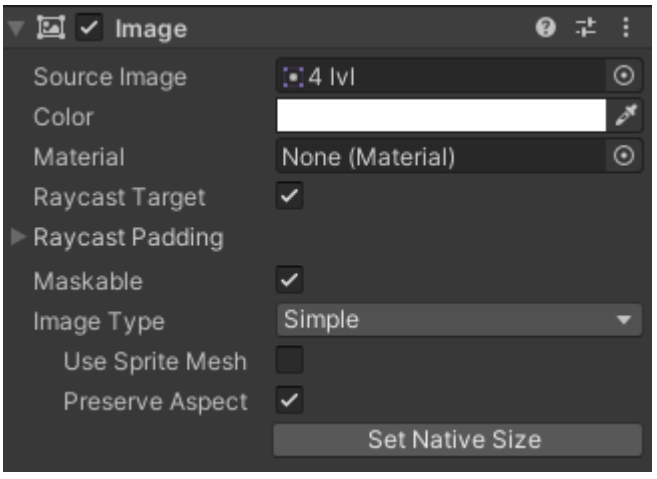

Рис. 2.8. Активація опції “Preserve Aspect” в компоненті Image

Для забезпечення коректного відображення об'єктів на екранах різних розмірів, необхідно зберегти відповідність їхніх пропорцій. Цього можна досягнути,

активувавши опцію “Preserve Aspect” в компоненті Image для об'єктів, які її містять (рис. 2.8).

Щоб об'єкти розташовувались на екрані відповідно до заданої орієнтації, ми повинні налаштувати параметр Anchor Presets для кожного візуального об'єкта на сцені. Це допоможе забезпечити стійкість позиціювання об'єктів незалежно від розміру екрана.

2.5 Збірка ігри на Android

Після того як ми оптимізували нашу гру потрібно зробити збірку на операційну систему Android. Для цього в нашій версії Unity є модуль Android Build Settings що імпортує JDK (Java Development Kit), NDK (Native Development Kit), SDK (Software Development Kit), Gradle які необхідні для збірки (рис. 2.9).

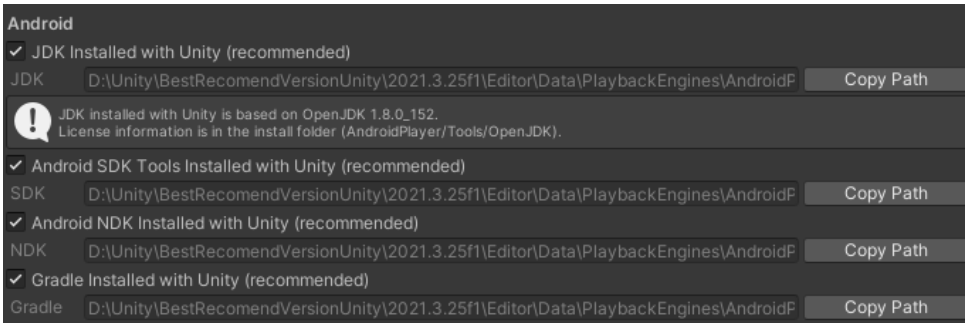

Рис. 2.9. Компоненти для збірки та шляхи до них

Відкривши панель Player Settings заповнюємо такі поля як Company Name та Product Name. Оскільки гра буде лише вертикальною, вибираємо у Defaul Orientation у випадаючому списку Portrait. В Other Settings вказуємо Package Name, заповнюємо поля Version та Bundle Version Code, а також виставляємо Minimum API Level та Target Api Level (рис. 2.10).

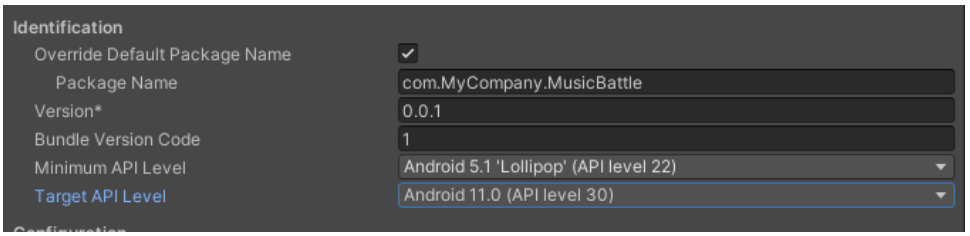

Рис. 2.10. Заповнення параметрів в панелі Player Settings

Для компіляції нашої гри в мобільний застосунок ми здійснюємо перенесення всіх ігрових сцен в розділ “Scenes in Build” (рис. 2.11). Важливо зауважити, що сцена з індексом нуль буде завжди першою яка відкривається при запуску додатка.

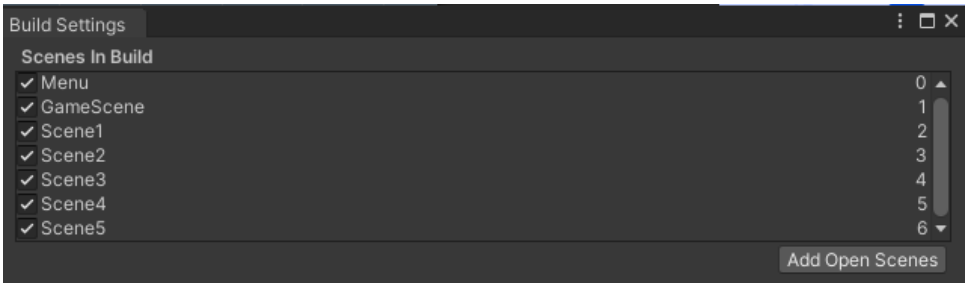

Рис. 2.11. Ігрові сцени в розділі “Scenes In Build”

Наступним кроком стає активація режиму “Development Build”. Цей режим дає можливість здійснювати налагодження та відстежувати потенційні проблеми під час розробки (рис. 2.12).

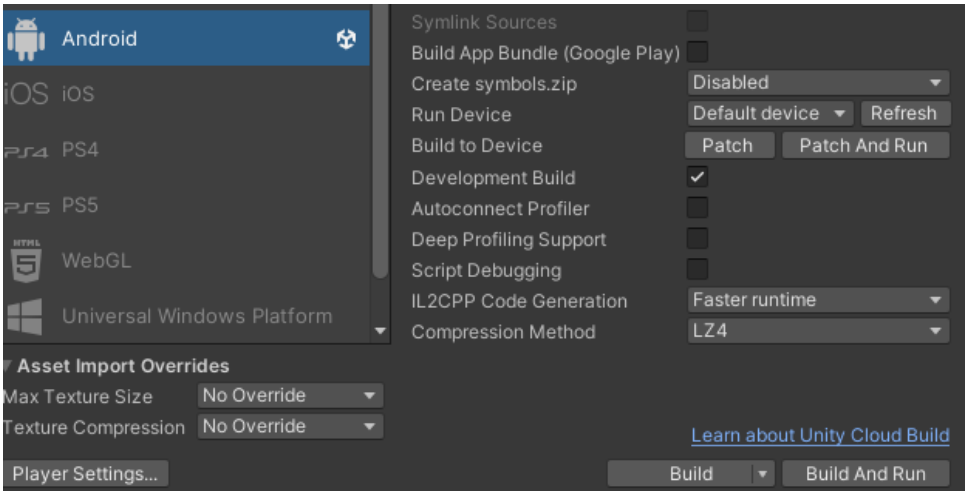

Рис. 2.12. Активація режиму “Development Build” в Unity

Після активації режиму “Development Build”, натискаємо кнопку “Build”. Процес компіляції може зайняти декілька хвилин, після яких ми отримуємо готовий APK-файл нашої гри, готовий до встановлення на мобільні пристрої з операційною системою android.

РОЗДІЛ 3. ОТРИМАНІ РЕЗУЛЬТАТИ

3.1 Тестування

Проведемо досліджений нами метод тестування Compatibility Testing. Рушій Unity дозволяє провести цей метод тестування безпосередньо у самій програмі. В Unity у вікні Game, зверху обираємо відкриваючий список і обираємо різні розмірності екрана і тестуємо застосунок (рис. 3.1). Переконуємось що він адаптивний до будь-якого розширення мобільного пристрою.

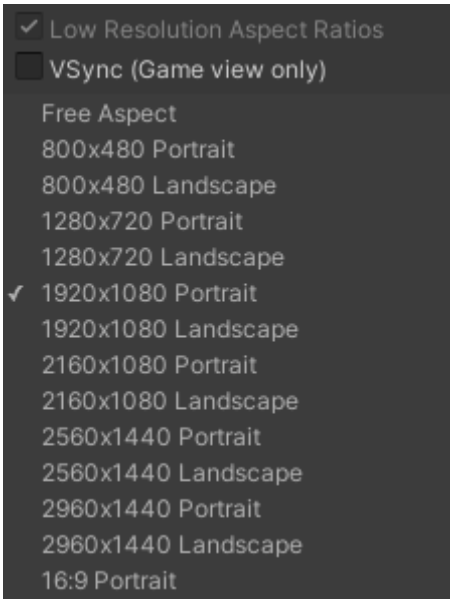

Рис. 3.1. Вибір розмірності для процесу тестування

Після завершення розробки гра була встановлена на мобільний пристрій, що працює на операційній системі Android для проведення фінального тестування. Запустивши додаток, ми переконуємося в його стабільності - гра не зазнає випадкових зупинок чи збоїв (рис. 3.2).

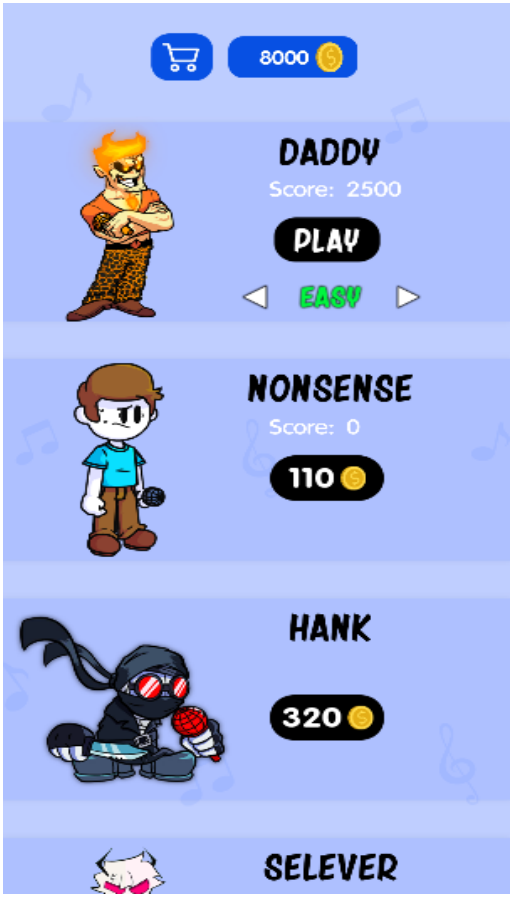

Рис. 3.2. Тестування стабільності гри на Android

У другому етапі перевірки ми зосереджуємося на відображенні графічних об'єктів. Всі візуальні елементи відтворюються належним чином, не виявлено дефектів у відображенні що свідчить про належну оптимізацію інтерфейсу (рис. 3.3).

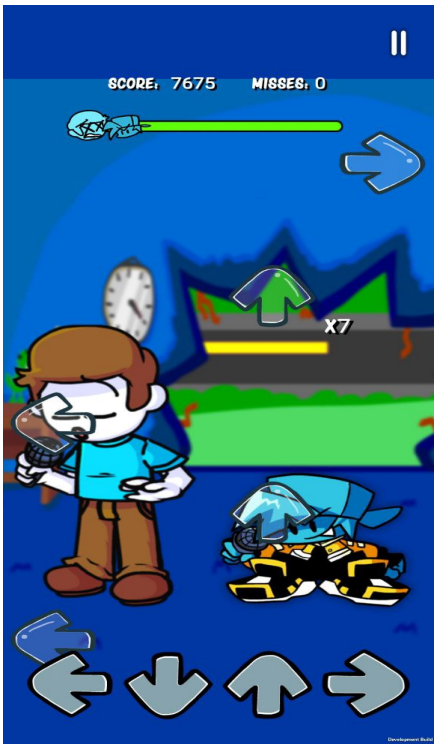

Рис. 3.3. Тестування графічних об’єктів гри на смартфоні

На третьому етапі ми переходимо до вивчення ігрової динаміки. Для прикладу включаємо 3 рівень і спостерігаємо за тим, як виконуються анімації, звукові ефекти та інші аспекти гри. Все працює відповідно до задуманого, без відхилень від планованих сценаріїв.

Наступним кроком є тестування ігрової механіки - накопичення ігрової валюти й покращення рекордів. Після кількох хвилин гри, ми успішно заробили ігрову валюту і побили попередні рекорди (рис. 3.4).

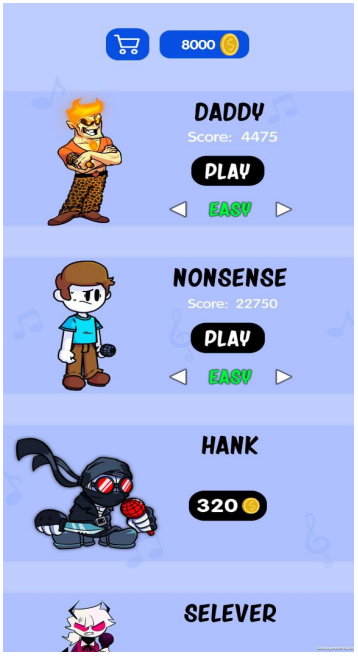

Рис. 3.4. Оцінка ігрової динаміки і механіки

В останньому етапі перевірки ми тестуємо збереження прогресу гри. Закривши та знову відкривши додаток, ми виявили, що всі зароблені монети та рекорди були належно збережені. Це свідчить про успішну реалізацію системи збереження стану гри. Таким чином, після докладного тестування на різних етапах, гра повністю готова до експлуатації.

3.2 Використання

Запустивши додаток ми бачимо користувацький інтерфейс. У нас є можливість вибрати складність до початкового рівні й почати його. Під час старту гри відбувається перехід на іншу сцену із самою грою. Передбачено що гру можна зупинити в будь-який момент натиснувши значок паузи, а далі продовжити або ж вийти в головне меню. Після початкових звуків, зверху над персонажами починають падати стрілки чотирьох кольорів, потрібно внизу натискати на чотири кнопки відповідних кольорів, при вдалому попаданню наш персонаж почне рухатись.

Після завершення грального процесу нам відкривається вікно з інформацією ігрового прогресу, відображається сума зароблених очок, кількість зірок і монети,

сума яких також залежить від вибраної складності й сумі ігрових очок. Також на цьому вікні розміщені дві кнопки: “Почати заново” та “Вийти в головне меню”. Після декількох сеансів, назбиравши певну суму монет, у нас з’являється можливість відкрити новий рівень, або ж купити нового персонажа в магазині. Гра вважається пройденою після того, як користувач розблокує усі рівні та персонажів.

ВИСНОВОК

В результаті виконання дипломної роботи розв’язано задачу програмної реалізації 2D мобільної гри за допомогою ігрового рушія Unity. При цьому були отримані такі результати:

- Досліджено сучасні методи створення ігор.
- Детально вивчено особливості роботи з ігровим рушієм Unity.
- Проведено дослідження методів тестування ігор.
- Розроблено користувацький інтерфейс, що є інтуїтивно зрозумілим і простим у використанні.
- Програмно реалізовано застосунок.
- Виконано оптимізацію застосунку для коректної роботи на мобільних пристроях.
- Зібрано гру під операційну систему Android.
- Проведено тестування гри, результати якого підтвердили високу якість кінцевого продукту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. DaGraca M., Lukosek G. Learning C# 7 by developing games with unity 2017 - third edition: learn C# programming by building fun and interactive games with unity. Packt Publishing - ebooks Account, 2017. 290 с.
2. Gee J. P. What Video Games Have to Teach Us About Learning and Literacy. Palgrave Macmillan, 2004. 240 с.
3. Geig M. Unity game development in 24 hours, sams teach yourself. Sams Publishing, 2013. 383 с.
4. Hocking J. Unity in Action: Multiplatform game development in C#. Manning Publications, 2018. 400 с.
5. Mednieks Z. Programming android. Sebastopol, Calif : O'Reilly, 2012. 542 с.
6. Schultz C. P. Game testing all in one. 2nd ed. Dulles, Va : Mercury Learning and Information, 2012. 434 с.
7. Unity documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.unity.com/.
8. Okita A. Learning C# Programming with Unity 3D. Routledge, 2014. 690 с.
9. Koster R. Theory of Fun for Game Design. O'Reilly Media, 2013. 300 с.
10. Nystrom R. Game Programming Patterns. Apress, 2011. 300 с.