ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

ПОЯСНЮВАЛЬНА ЗАПИСКА

до бакалаврської роботи

на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА ГРИ "TRAFFIC JUMBLE" У ЖАНРІ HYPER CASUAL З ВИКОРИСТАННЯМ ІГРОВОГО РУШІЯ UNITY ПІД МОБІЛЬНУ ПЛАТФОРМУ ANDROID МОВОЮ C#»**

Виконав: студент 4 курсу, групи ПД – 42
спеціальності:
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

Осьмак В.Р.
(прізвище та ініціали)

Керівник Дібрівний О.А.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

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

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ О.В. Негоденко

«____» _____2023 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

ОСЬМАКА ВЛАДИСЛАВА РОМАНОВИЧА
(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка гри "Traffic Jumble" у жанрі hyper casual з використанням ігрового рушія unity під мобільну платформу android мовою C#»

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

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

2. Строк подання студентом роботи « 1 » червня 2023 року

3. Вихідні данні до роботи:

3.1 Науково-технічна література пов'язана з темою щодо розробки ігор.
3.2 Офіційна документація Unity
3.3 Офіційна документація Rider
3.4 Існуючі ігри жанру Hyper Casual
3.5 Підходи до побудови ігор

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

4.1 Аналіз предметної області
4.2 Аналіз та вибір засобів реалізації
4.3 Проектування гри
4.4 Розробка гри
4.5 Тестування гри
4.6 Висновки

5. Перелік демостраційного матеріалу:

5.1. Титульний слайд
5.2. Мета, об’єкт та предмет дослідження
5.3. Порівняння аналогів
5.4. Технічне завдання
5.5. Програмні та технічні засоби реалізації
5.6. Use case діаграма
5.7. Діаграми класів
5.8. Інтерфейс користувача
5.9. Тестування
5.10.Апробація результатів дослідження
5.11.Висновки

6. Дата видачі завдання « 25 » лютого 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	25.02.23 – 27.02.23	Виконано
2	Дослідження аналогів та актуальності додатку	02.03.2023	Виконано
3	Аналіз та вибір інструментів для розробки додатку	05.03.2023	Виконано
4	Проектування та реалізація	29.04.2023	Виконано
5	Вступ, висновки, реферат	12.05.2023	Виконано
6	Розробка презентації	14.05.2023	Виконано
7	Попередній захист	24.05.2023	Виконано
8	Здача роботи	01.06.2023	

Студент Осьмак В.Р.

(підпис) (прізвище та ініціали)

Керівник роботи Дібрівний О.А.

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи: 68 стр., 26 рис., 1 дод., 15 джерел

UNITY, C#, HYPER CASUAL , ГРА, ВІДЕОІГРИ, ANDROID.

Об’єкт дослідження – ігровий процес у жанрі "hyper casual" під платформу Android.

Предмет дослідження – гра в жанрі "hyper casual" під платформу Android.

Мета роботи – підвищення цікавості гемплею за рахунок використання гіперказуальних механік.

Наукова новизна – створення нових та перетворення відомих ігрових механік, притаманних для ігор жанра hyper casual; створення інтуїтивного та привабливого дизайну гри, що зачепить гравця цікавий досвід.

Методи дослідження – Методи проектування та розробки програмного забеспечення в контексті розробки ігор під мобільні платформ. Методи аналізу відкритих інформазійних джерел. Методи тестування додатків.

Галузь використання – Гра може бути використана як спосіб ненадовго відволіктися та витратити час на нескладний процес.

ЗМІСТ

ВСТУП

Актуальність теми. На сьогоднішній день в сучасному світі розвиток мобільних пристроїв та ігрової індустрії зростає з кожним днем. За даними багатьх інфо ресурсів обсяги доходів від мобільних ігр зростають з кожним роком. Користувачі віддають перевагу іграм, які можна грати в будь-який час та в будь-якому місці. Саме це і є ігри жанру hyper casual відрізняються від інших жанрів своєю простотою та легкістю. Вони не потребують складних налаштувань та розуміння правил гри, що дозволяє гравцям одразу ж підключатися до гри та насолоджуватися процесом.

Гіпер-кежуал ігри є одним з найбільш популярних жанрів серед користувачів мобільних пристроїв. Ігри цього жанру мають простий геймплей та контроль, що дозволяє швидко та легко засвоювати їх користувачам. Такі ігри зазвичай мають невисокий рівень складності, але водночас є захоплюючими та можуть забезпечити години цікавої гри.

Застосування ігрового рушія Unity для розробки такої гри дозволяє значно полегшити процес розробки та оптимізувати його для мобільних пристроїв з операційною системою Android. Також Unity надає можливість розробникам використовувати готові модулі та рішення, що дозволяє значно збільшити продуктивність розробки.

Об’єкт дослідження – ігровий процес у жанрі "hyper casual" під платформу Android.

Предмет дослідження – гра в жанрі "hyper casual" під платформу Android.

Мета роботи - підвищення цікавості гемплею за рахунок використання гіперказуальних механік.

Наукова новизна проекту – створення нових та перетворення відомих ігрових механік, притаманних для ігор жанра гіперкежуал; створення інтуїтивного та приваблвого дизайну гри, що зачепить гравцю цікавий досвід.

У дипломному проекті був проведений аналіз ринку мобільних ігор та додатків-аналогів. Проаналізовано та виявлено переваги і недоліки програмних інструментів розробки. Проаналізовано особливості розробки мобільних ігор жанра Hyper Casual.

Гра розроблена в ігровому двигуні Unity, програмний код написаний в середовищі розробки Rider на мові програмування C#.

Завдання дослідження:

1. Аналіз відеоігор і їх жанрів
2. Аналіз ринку гіперказуальних ігор для мобільних платформ.
3. Вивчення можливостей ігрового рушія Unity.
4. Проектування геймплею гри.
5. Пошук графіки та дизайну гри.
6. Проектування архітектури гри
7. Розробка гри.
8. Тестування гри та виявлення помилок.
9. Аналіз результатів роботи.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Відеоігри

Відеоігри - це електронні системи, з якими можна взаємодіяти за допомогою пристроїв введення, таких як контролер, клавіатура або джойстик. Відеоігри можуть використовуватися для розваг і відпочинку, але також можуть бути використані для змагань і для навчання. Деякі відеоігри розроблені для поліпшення моторики і руко-око координації.

Перші відеоігри були прототиповані в 1950-х та 1960-х роках, а до 1970-х виникла ціла індустрія навколо відеоігор. Насправді, було так багато відеоігор, що в 1983 році галузь переживала кризу через занадто велику кількість ігор низької якості.

Ця криза змусила розробників і видавців відеоігор дуже обережно вибирати, які ігри вони випускають, і протягом 2000-х років більшість відеоігор робили великі компанії. Однак з появою інтернету малі розробники почали створювати свої власні ігри, які називаються інді ігри.

Протягом багатьох років відеоігри вважалися хобі або відпочинком, але наприкінці 2000-х років почали набирати популярності електронні змагання. Електронні змагання, або Esports, це змагання між професійними гравцями відеоігор.

1.2 Походження відеоігор

Перші відеоігри були створені ще у 1950-х роках, коли ьвчені створювали прості електронні ігри, щоб перевірити можливості комп'ютерів. Дуже прості ігри, такі як "хрестики-нулики", могли бути запрограмовані в ранніх комп'ютерах, і використовуватися як демонстрації, щоб показати потужність комп'ютерів. Одним з менш відомих фактів про відеоігри є те, що багато ранніх розробників не думали,

що вони будуть популярні, і використовували їх лише для тестування можливостей комп'ютера.

Перша відеоігра, що була розповсюджена, була розроблена протягом 1961-1962 років і називалася "Spacewar!". Вона мала дуже просту графіку і гемплей де два гравці могли симулювати бойові дії у космосі. У 1970-х роках програмісти почали створювати мови програмування, які полегшили людям створення програм, включаючи відеоігри. Чим більше програмістів почали розробляти свої власні ігри, тим більше людей почали бачити потенціал у іграх.

Це прискорило швидкість розвитку відеоігор, і передовсім вперше були створені та розповсюджені аркадні відеоігри та перші домашні відеоігрові консолі. Аркади раніше складалися з механічних ігор, таких як кран-машина, але протягом цього десятиліття відеоігри стали займати провідні позиції.

1.3 Розквіт відеоігор

Першою успішною аркадною відеоігрою стала гра Pong, яка була випущена в 1972 році. Ця проста гра дозволяла двом гравцям змагатися в грі в настільний теніс, а результати відображалися у верхній частині екрана. Хоча за сучасними стандартами вона здається простою, на той час ця гра була революційною.

Із поширенням новин та фактів про відеоігри, із зростанням розмірів промислу, програмісти у Великій Британії, США, Росії, Японії та багатьох інших країнах почали створювати свої власні відеоігри. Деякі програмісти взяли основні ідеї однієї гри та скопіювали її, додавши нові елементи. Інші експериментували з абсолютно новими концепціями, деякі з яких стали популярними хітами. Формат настільного тенісу Pong був відтворений багато разів, але інші типи ігор, такі як стрілялки та пригодницькі ігри, стали ще більш популярними.

Це призвело до величезного буму у виробництві та продажу відеоігор впродовж 1970-х та на початку 1980-х років, який називається "золотим віком" аркадних відеоігор. Деякі з найбільш іконічних та популярних відеоігор були

створені в цей період, включаючи Space Invaders, Pac-Man, Asteroids, Breakout, Galaga та Donkey Kong

1.4 Типи відеоігор за пристроями

Існує безліч різних жанрів відеоігор, але спочатку проаналізуємо різні типи пристроїв для ігор. Зоокрема виділимо основні з них:

- Відеоігри для ПК
- Консольні відеоігри
- Мобільні відеоігри

1.4.1 Відеоігри для ПК

Комп’ютерні ігри є видом інтерактивного розважання, що запускаеться на персональних комп’ютерах. Ці ігри дозволяють користувачам зануритися в віртуальний світ і взаємодіяти з ним, виконуючи різноманітні завдання та дії за допомогою клавіатури, миші або інших пристроїв введення.

Комп’ютерні ігри були розроблені в 1960-х роках і були доступні на великих комп’ютерах, таких як IBM mainframes. З тих пір цей вид розваг поширився по всьому світу та став доступним на різних платформах, включаючи персональні комп’ютери, консолі для ігор, смартфони та планшети.

Комп’ютерні ігри можуть бути різних жанрів, таких як стратегії, шутери, рольові ігри, гонки та імітації. Багато з них мають складну історію та персонажів, що розвиваються з часом, і дозволяють гравцеві вибирати різні шляхи та рішення, що впливають на подальші події гри.

Комп’ютерні ігри мають великий вплив на культуру та розвиток технологій. Вони часто стають предметом досліджень у галузі психології, соціології та інших наук, що вивчають вплив розважальної індустрії на суспільство та культуру. Також комп’ютерні ігри стали важливим елементом відеоігрової індустрії, що оцінюється в мільярди доларів.

1.4.2 Консольні відеоігри

Консольні відеоігри - це вид відеоігор, які розробляються та граються на консолях, спеціально створених для цього. Консолі можуть бути переносними або настільними, тобто можуть бути підключені до телевізора чи комп'ютерного монітора. Консольні ігри можуть бути одиночними або мультиплеєрними, тобто грати можна як самостійно, так і з іншими гравцями.

Консольні відеоігри виникли в 1970-х роках із запуском перших ігрових консолей, таких як Magnavox Odyssey та Atari. З тих пір було розроблено безліч консолей різних поколінь, таких як Nintendo Entertainment System (NES), Sega Genesis, PlayStation, Xbox, Nintendo Switch та інші. Кожна консоль має свою унікальну архітектуру, яка впливає на графіку, звук, контролери та інші аспекти гри.

Консольні ігри можуть бути різних жанрів, включаючи екшн, пригодницький, стратегію, спортивний та багато іншого. Оскільки консолі мають обмежену апаратну конфігурацію, розробники часто оптимізують гри під конкретну консоль, що дозволяє отримати кращу графіку та швидкодію. Крім того, консольні ігри можуть мати ексклюзиви, тобто гри, які доступні тільки на конкретній консолі. Це є однією з причин, чому гравці можуть купувати кілька консолей.

У сучасному світі консольні ігри продовжують залишатись популярними. Найбільш відомі консолі на сьогоднішній день - це PlayStation 5, Xbox Series X/S та Nintendo Switch. Крім того, деякі консолі, такі як Nintendo Switch, можуть використовуватись як переносні ігрові пристрої. Переносність є однією з головних переваг консолей Nintendo Switch і PlayStation Vita. Консоль Nintendo Switch має два режими: телевізійний і портативний. У портативному режимі консоль може бути використана як планшет зі знімними джойстиками, або як портативна консоль з прикріпленими джойстиками. Це дозволяє гравцям грати відеоігри в будь-який час і в будь-якому місці, не потребуючи доступу до телевізора або монітора.

Крім того, консолі Nintendo Switch і PlayStation Vita мають багато ігор, які були створені спеціально для переносного геймінгу. Ці ігри зазвичай мають більш

прості управління, меншу кількість діалогів та деталей на екрані, що дозволяє гравцям швидко переключатися між різними іграми і грати в них на ходу. Деякі ігри також мають можливість мультиплеєра, що дозволяє гравцям грати з друзями на тому ж самому пристрої.

Консольні ігри також часто мають кращу графіку та звук, ніж мобільні ігри. Це пов'язано з тим, що консолі мають більше потужності і можуть відтворювати більш складні графічні ефекти та звукові доріжки. Крім того, ігри на консолях зазвичай мають більшу кількість контенту, що дозволяє гравцям грати в одну гру протягом більш тривалого часу, не надто повторюючи вже пройдені рівні.

1.4.3 Мобільні відеоігри

Мобільні ігри - це ігри, які можна грати на мобільних пристроях, таких як смартфони або планшети. Вони стали дуже популярними останнім часом завдяки широкому поширенню мобільних пристроїв серед користувачів з усього світу.

Особливість мобільних ігор полягає в їхній доступності та простоті використання. Більшість з цих ігор можна завантажити безкоштовно з мобільного додатку або магазину додатків, а деякі з них можна купити за невелику ціну або здобути спеціальні предмети або функції за гроші.

Мобільні ігри можуть мати різні жанри, включаючи стратегію, головоломки, рольові ігри, екшн, гонки та інші. Більшість з них розробляється з урахуванням особливостей мобільних пристроїв, таких як сенсорний екран або можливість грати в мережі Інтернет.

Незалежно від того, чи ви шукаєте розвагу для короткого перериву в роботі або бажаєте грати улюблену гру, коли ви не вдома, мобільні ігри можуть бути відмінним варіантом. Вони доступні, легкі використанні та можуть забезпечити безліч годин розваги.

1.5 Ринок мобільних ігор та місце HYPER CASUAL в ньому

Ринок мобільних ігор - один з найбільших і швидко зростаючих сегментів ринку розваг. За даними дослідницьких компаній, таких як App Annie, у 2022 році глобальний ринок мобільних ігор досяг вартості $184,4 мільярдів доларів, що становить майже 50% від загальної вартості глобального ринку ігор. Прогнозується, що цей ринок продовжуватиме зростати в майбутньому.

У мобільних іграх існує багато різних жанрів, включаючи пригодницькі, аркадні, стратегічні та багато інших. Однак, серед них гіперказуальні ігри займають все більше місця на ринку. Це обумовлено їх простотою та доступністю, що дозволяє залучати більше гравців, а також високою рентабельністю для розробників.

Гіперказуальні ігри зазвичай мають просту механіку та геймплей, що дозволяє гравцям швидко засвоювати їх і відчувати задоволення від гри. Такі ігри, як Subway Surfers, Temple Run 2, та Car Run 2, стали популярними завдяки своїй простоті та залучаючій геймплейній механіці.

За останні роки було створено баг ато гіперказуальних ігор, які зайняли своє місце на ринку мобільних ігор. Вони стали досить популярними завдяки своїй доступності та легкості в освоєнні. На даний момент, гіперказуальні ігри стали одним з найбільш доходних жанрів на ринку мобільних ігор, заробляючи мільйони доларів щороку.

Найбільш відомі гіперказуальні ігри, такі як "Flappy Bird" та "Candy Crush Saga", допомогли відкрити цей жанр для гравців і розробників. Вони показали, що ігри, що не вимагають складних графічних ефектів та великої кількості часу на освоєння, можуть бути дуже популярними. Нові розробники можуть використовувати ці ігри як зразок для створення власних гіперказуальних ігор.

Ще одна перевага гіперказуальних ігор - це їх великий потенціал для монетизації. Багато гіперказуальних ігор безкоштовні для завантаження, але містять внутрішні покупки, рекламу або платні підписки. Це дає змогу

розробникам заробляти гроші на своїх іграх, навіть якщо вони безкоштовні для гравців.

Проте, зростаюча кількість гіперказуальних ігор на ринку також призводить до збільшення конкуренції. Щоб привернути увагу гравців, розробники повинні створювати ігри, які не тільки легкі для освоєння, але й цікаві та залучаючі. Також важливо мати якісну рекламу та ефективну стратегію монетизації.

Загалом, гіперказуальні ігри стали дуже важливою частиною ринку мобільних ігор, оскільки вони пропонують новий формат, який дозволяє створювати веселі та прості ігри зі складовими ігрових жанрів, таких як аркади, головоломки, пригодницькі ігри тощо.

Цей жанр мобільних ігор є досить молодим, проте він швидко розвивається та набуває популярності серед гравців. Багато розробників використовують технології машинного навчання, щоб створювати персоналізовані ігри з використанням даних гравця та відстеженням їх поведінки в грі.

Оскільки гіперказуальні ігри не вимагають великих інвестицій, це дозволяє багатьом розробникам створювати свої власні ігри та займати своє місце на ринку. Також це забезпечує здешевлення вартості розробки та підтримки ігр.

Незважаючи на те, що гіперказуальні ігри вважаються більш простими в порівнянні з іншими жанрами, вони все ж мають свої вимоги до якості та рівня геймплею. Розробники гіперказуальних ігор повинні приділяти особливу увагу розробці ігрових механік, звуковому дизайну та візуальному оформленню гри.

Оскільки гіперказуальні ігри зазвичай мають простий сценарій та ігровий процес, їх можна легко переносити на різні платформи, такі як ПК, консолі та інші мобільні пристрої. Це забезпечує розробникам можливість розширювати свої аудиторії та залучати нових гравців.

У світі мобільних ігор гіперказуальні ігри займають значну нішу і стали невід’ємною частиною ринку. Вони є популярними серед широкої аудиторії, включаючи як чоловіків, так і жінок різного віку. Цей тип ігор набуває все більшої популярності завдяки своїй легкості та доступності, які відповідають вимогам багатьох гравців.

На сьогоднішній день гіперказуальні ігри продовжують розвиватися і покращуватися. З’являються нові ідеї, геймплей, графічні та звукові ефекти, які дозволяють гравцям насолоджуватися грою і отримувати нові враження. Розвиток технологій також допомагає створювати все більш складні та захоплюючі ігри.

У майбутньому гіперказуальні ігри, ймовірно, й надалі продовжать займати важливу нішу на ринку мобільних ігор. Розробники будуть працювати над поліпшенням геймплею, графіки та звуку, а також збільшенням масштабу гри та функціональності. Крім того, гіперказуальні ігри можуть стати більш соціальними за допомогою інтеграції з соціальними мережами та іншими сервісами, такими як Google Play Games або Apple Game Center. Це дасть гравцям можливість зв'язуватися з іншими гравцями та обмінюватися ігровим досвідом.

1.6 Жанри мобільних ігор

На ринку мобільних ігор представлено безліч різноманітних жанрів на будь-який смак. Проаналізуємо деякі із них:

- Пригодницький
- Аркадни
- Казуальний
- Королівська битва
- Головоломка
- Словесні
- Гіпер-казуальні

1.6.1 Пригодницькі ігри

У таких іграх гравець виконує роль головного героя, який вирушає на небезпечні та цікаві пригоди з метою врятувати світ або виконати якісь завдання. Відомі пригодницькі ігри на мобільних пристроях, такі як Tomb Raider, Assassin's Creed та Uncharted, є класичними представниками цього жанру.

Основним елементом гри є створення непередбачуваних ситуацій, які вимагають від гравця логічного мислення та швидкого реагування. Часто у пригодницьких іграх присутні загадки та головоломки, що додає грі додаткової складності. Графіка та звук таких ігор зазвичай на високому рівні, а відтворення різних країв та сценаріїв надають гравцеві почуття присутності в грі.

Зазвичай пригодницькі ігри на мобільних пристроях мають досить довгу історію або складний сюжет, що дозволяє гравцеві відчути себе частиною епічного пригодницького фільму. Крім того, в таких іграх є можливість розвивати персонажа, вдосконалюючи його навики та характеристики, що дає гравцеві більше контролю над грою.

Одним з головних факторів успіху пригодницьких ігор на мобільних пристроях є їх доступність. Більшість таких ігор можна завантажити з магазинів додатків безкоштовно або за невелику ціну. Крім того, вони не потребують потужних характеристик пристрою, тому їх можна грати на практично будь-якому.

1.6.2 Аркадні ігри

Це тип ігор, які мають простий геймплей, високу швидкість дії та вимагають від гравця швидкісного реагування на події на екрані. Головна мета гравця в аркадних іграх - набирати якомога більше очок або збирати різні бонуси. Ігри цього жанру зазвичай не мають складного сюжету, а зосереджені на забезпеченні максимальної кількості задоволення від швидкої та екшн-наповненої гри.

У світі аркадних ігор зустрічаються дуже різноманітні ігри зі своїми власними правилами та геймплеєм. Найбільш популярними серед аркадних ігор є різноманітні аркадні гонки, такі як Need for Speed, Asphalt, або Hill Climb Racing. Ці ігри мають простий геймплей, який полягає в керуванні машинами та перегонами на різних трасах. Окрім цього, до аркадних ігор можна віднести багато ігор, в яких головним завданням є виживання, таких як Doodle Jump або Temple Run. В таких іграх гравцеві потрібно діяти швидко та точно, щоб уникати перешкод і збирати бонуси на своєму шляху.

1.6.3. Казуальні ігри

Казуальний жанр мобільних ігор орієнтується на широку аудиторію, що любить грати в прості та легкі ігри. Це можуть бути головоломки, аркади, ігри на швидкість реакції, різноманітні ігри з кольорами, героями, звірятами, рослинами та іншими елементами.

Одна з головних особливостей казуальних ігор полягає в їхній простоті та зрозумілості. Вони зазвичай не мають складних історій, складних керувань, глибоких стратегій або важких завдань. Це дозволяє гравцеві з легкістю засвоювати гру та отримувати від неї задоволення.

До казуальних ігор також належать тайм-кіллери, тобто ігри, які можна грати у вільний час, наприклад, під час очікування на автобус або в черзі. Ці ігри часто мають відкрите геймплей та можуть бути безкінечними, дозволяючи гравцеві відчувати виклик та соревноваться з собою та іншими гравцями.

Казуальні ігри також зазвичай мають простий та дитячий дизайн, що робить їх привабливими для дітей та підлітків. Вони можуть бути яскравими, кольоровими та миленькими, що дозволяє гравцеві розслабитися та насолоджуватися процесом гри.

До популярних казуальних ігор можна віднести такі гри, як Candy Crush, Angry Birds, Subway Surfers, Fruit Ninja, Temple Run та інші. Вони успішно займають верхні місця в рейтингах мобільних додатків та мають велику аудиторію фанатів по всьому світу.

1.6.4 Королівська битва

В таких іграх гравці змагаються між собою на великому полі бою, під час якого потрібно виживати, збирати ресурси та знищувати інших гравців. Особливістю цього жанру є те, що гравці з початку гри починають без зброї та ресурсів, тому їм доводиться шукати їх на карті.

Один з головних елементів королівських битв - це рандомність. На кожній карті знаходяться випадкові предмети, зброя та інші ресурси, які гравці повинні шукати. Крім того, на карту періодично падають коробки з літаків, які містять рідкісну зброю та інші ресурси. Гравці повинні добре відчувати момент і знати, коли їм потрібно ризикувати та забрати ці коробки, щоб отримати перевагу над іншими гравцями.

Іншою важливою складовою королівських битв є розмір карти та кількість гравців. Зазвичай, картіи бувають досить великі, щоб забезпечити достатньо простору для переміщення гравців та інших елементів гри. Щодо кількості гравців, то вона може коливатися від декількох до декількох десятків гравців на одній карті.

Найпопулярнішою королівською битвою на мобільних платформах є PUBG Mobile. У грі гравці зійдуться в боротьбі на великому острові та будуть шукати зброю та інші ресурси, щоб залишитися в живих. Крім того, у грі є можливість створити команду з друзями та грати в режимі "бойових груп".

1.6.5 Головоломки

Цей жанр ігор передбачає вирішення різноманітних завдань, які потребують міркувань та розв’язання логічних задач. Головоломки можуть бути вигадливими та складними, тож їх вирішення може вимагати багато часу та зосередження.

Одним із видів головоломок є класичні пазли. Гравці повинні скласти картинку з декількох частинок, що можуть бути різних форм та розмірів. Для досягнення мети гравці повинні знайти правильне положення кожної частинки та скласти їх в правильному порядку.

Іншим видом головоломок є ігри, де гравці повинні знайти вихід з лабіринту або розв’язати завдання з переміщенням об’єктів в певному порядку. Такі ігри дозволяють розвивати не тільки логічне мислення, а й здібності до просторового мислення.

До головоломок можна також віднести ігри з використанням фізики. У таких іграх гравці повинні користуватись знаннями фізики для розв’язання завдань, таких як розбиття скляної вази на частинки чи розкриття складного механізму.

Останнім часом все більш популярними стають ігри, що передбачають розв’язання різноманітних головоломок з використанням елементів віртуальної реальності. У таких іграх гравець отримує змогу зануритися в неповторний світ, де йому належить вирішувати різні завдання та головоломки відповідно до правил цього світу.

1.6.6 Словестні

Цей жанр ігор передбачає взаємодію гравця з персонажами та сюжетом за допомогою текстових повідомлень. Одним із головних переваг цього жанру є можливість взаємодії з ігрою в будь-який час і з будь-якого місця з використанням мобільного телефону або планшета.

Словестні мобільні ігри відрізняються від традиційних комп'ютерних ігор тим, що вони більш зосереджені на історії та розвитку персонажів. Головним завданням гравця є взаємодія з ігровим світом та розв'язання різноманітних завдань, які можуть бути пов'язані з розгадуванням загадок, збором предметів, прийняттям рішень та вирішенням історійних конфліктів.

Одним із головних чинників популярності цього жанру є його доступність для широкого кола користувачів. Більшість словестних ігор мають простий та зрозумілий інтерфейс, що дозволяє гравцям легко зрозуміти, як взаємодіяти з ігровим світом та персонажами. Крім того, словестні ігри дозволяють гравцям зануритися у світ історії та фантастики, що робить цей жанр особливо привабливим для фанатів книжок та кіно.

У загальному, словестні мобільні ігри є чудовим варіантом для тих, хто любить історії, пригоди та загадки, а також для тих, хто шукає відповідний спосіб розважитися та розслабитися у будь-який час і з будь-якого місця.

1.6.7 Гіпер-казуальні ігри

Жанр гіперкажуал мобільних ігор - це розважальний жанр, що поєднує в собі простоту і доступність для користувачів. Ігри цього жанру зазвичай мають надзвичайно прості правила та механіку гри, що дає змогу гравцям легко і швидко освоювати їх і шукати в них задоволення.

Ще однією характеристикою гіперкажуал ігор є їхній відносно короткий час гри, що часто не перевищує декількох хвилин. Це дає змогу гравцям залучатися до гри в будь-який момент вільного часу та з легкістю переривати гру, якщо з'явилася необхідність.

Також, гіперкажуал ігри часто мають просту графіку та звукове супроводження, що дозволяє їм займати невеликий обсяг пам'яті на мобільних пристроях та працювати на майже будь-яких платформах.

Останнім часом гіперкажуал ігри стали надзвичайно популярними в мобільному геймінгу, зазвичай займаючи верхні позиції в рейтингах додатків у магазинах додатків. Це пояснюється тим, що вони дозволяють користувачам займатися геймінгом на шляху до роботи, в перервах між заняттями або відпочинком вдома.

1.7 Аналіз існуючих розробок

Нижче проводиться аналіз та опис популярних ігор жанру Hyper Casual подібних до розроблюваної гри "Traffic Jumble”.

1.7.1 Subway Surfers

Один з основних елементів Subway Surfers – це гра на швидкість та рефлексы. Гравець повинен бути дуже швидким та спритним, щоб уникнути поліції та перешкод, які з'являються на його шляху. Він може виконувати різноманітні трюки, такі як розвороти на 360 градусів та скейтбордінг, щоб отримати додаткові бали.

Subway Surfers також пропонує різноманітні місії та завдання, які гравець може виконувати, щоб отримати додаткові бали та монети. Ці місії можуть бути

дуже різноманітними - від збору певної кількості монет до проходження певного шляху під певними умовами.

Гра має яскраві та кольорові локації, що допомагає створювати атмосферу міста, через яке проходить потяг. Крім того, гра має різноманітних персонажів та скейтборди, які можна відкривати та використовувати.

Рисунок 1.1 - Демонстрація геймплею в Subway Surfers

1.7.2 Temple Run 2

Один з основних елементів Temple Run 2 – це теж швидкість та рефлексы. Гравець повинен бути дуже швидким та спритним, щоб уникнути перешкод та пройти якомога далі. Він може виконувати різноманітні трюки та комбінації рухів, щоб отримати додаткові бали та бонуси.

Temple Run 2 також пропонує різноманітні місії та завдання, які гравець може виконувати, щоб отримати додаткові бали та монети. Ці місії можуть бути дуже різноманітними - від збору певної кількості монет до проходження певного шляху під певними умовами.

Одним з головних переваг Temple Run 2 є його графіка та дизайн. Гра має яскраві та деталізовані локації, що допомагає створювати атмосферу джунглів та

давніх храмів. Крім того, гра має різноманітні персонажі та обладнання, які можна відкривати та використовувати.

Рисунок 1.2 - Демонстрація геймплею в Temple Run 2

1.7.3 Car Run 2

Це безкоштовна мобільна гра в якій гравець виконує роль водія автомобіля, який повинен проїхати якомога більше дистанції, уникнувши перешкод на дорозі та збираючи бонуси. Гра доступна для пристроїв на базі операційної системи Android.

У грі Car Run 2 вам належить керувати автомобілем на швидкій дорозі, уникаючи зіткнень з іншими авто, вантажівками, автобусами та іншими перешкодами. У грі доступні різні типи автомобілів з різними характеристиками, від швидких спортивних машин до вантажівок з високою міцністю. Гра має кольорову та динамічну графіку, що дозволяє вам насолоджуватися грою на повну потужність

Окрім того, у грі Car Run 2 є система бонусів, яка додає додаткові можливості гравцеві. Зібравши певну кількість бонусів, гравець може отримати можливість

перетворити свій автомобіль на нерозбитний, тим самим, захистивши його від пошкоджень від зіткнень з іншими автомобілями.

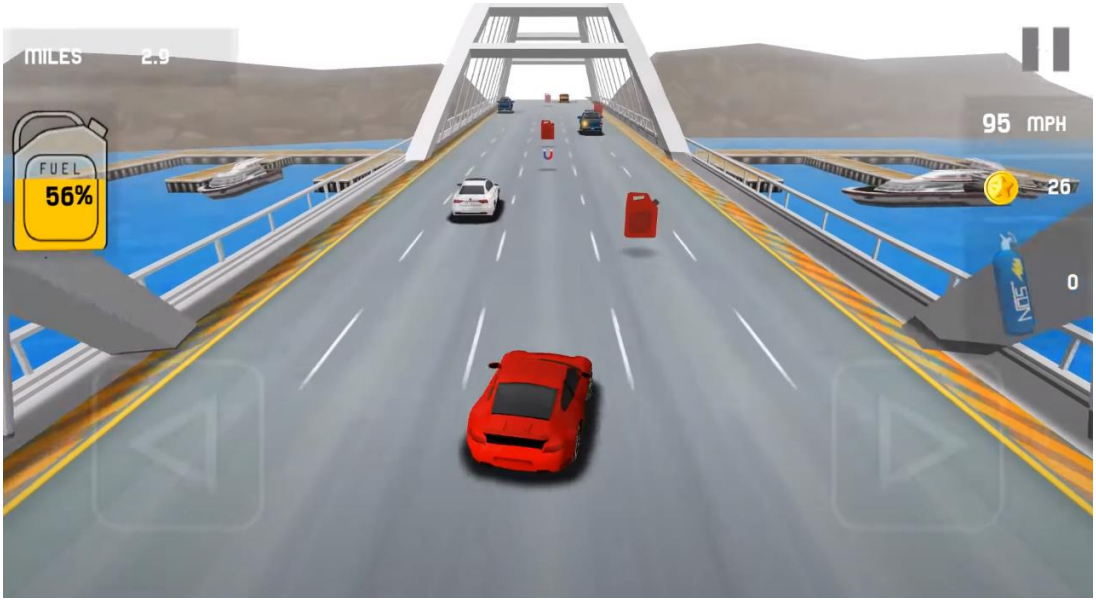


Рисунок 1.3 - Демонстрація геймплею в Car Run 2

Усі накопичені бонуси гравець може використовувати для покращення свого автомобіля та отримання нових можливостей для більш вдалої гри. Загальний рівень складності гри зростає поступово, що дозволяє гравцеві відчувати наростаючу складність і виклик.

1.7.4 2D Car Runner

Це мобільна гра в якій гравцеві потрібно керувати автомобілем, уникати перешкод і збирати бонуси. Гра має двомірну графіку та використовує візуальні ефекти для створення швидкості та динаміки на трасі.

У грі є кілька режимів гри, включаючи кампанію, безкінечний режим та мультиплеєр. Кампанія складається з декількох рівнів, в яких гравець повинен доїхати до фінішу за максимально короткий час та збирати бонуси на шляху. У безкінечному режимі гравець змагається з часом, щоб дійти якомога далі. У мультиплеєрі гравці можуть змагатися один з одним на тій же трасі.

Гра має різні типи автомобілів, які можна розблокувати за допомогою бонусів або грошей, які гравець заробляє під час гри. Кожен автомобіль має унікальні властивості, такі як швидкість та керованість, що можуть вплинути на гру.

У грі є також система досягнень, які гравець може отримати, виконуючи певні завдання або досягнення в грі. Це додає більше глибини та виклику до гри, оскільки гравець може спробувати здійснити всі досягнення.

2D Car Game це дуже проста та легко доступна гра, яка може бути цікавою для всіх вікових категорій.

Рисунок 1.4 - Демонстрація геймплею 2D Car Runner

1.8 Висновки до розділу

Підводячи підсумки до першого розділу даної дипломної роботи було проаналізовано предметну область відеоігор та їх типів в залежності від пристрою, на якому вони граються. Детально розглянуто походження та розквіт відеоігор, що дозволяє зрозуміти їхню історію та розвиток.

У розділі також досліджено ринок мобільних ігор, а також місце, яке займають в ньому гіпер-казуальні ігри. Вони стали дуже популярними та займають важливе місце на ринку мобільних ігор. Це зумовлено їхніми простими правилами та інтуїтивним інтерфейсом, що дозволяє залучати нових гравців та розширювати аудиторію.

У розділі також розглянуто різні категорії мобільних ігор, такі як пригодницькі, аркадні, казуальні, королівська битва, головоломки, словестні та гіпер-казуальні ігри. Кожна з них має свої унікальні особливості та правила.

У заключному розділі було проведено аналіз деяких існуючих розробок подібних до розроблюваної гри “ Traffic Jumble ”, зокрема Subway Surfers, Temple Run 2, Car Run 2 та 2D Car Game.

Для розробників гіпер-казуальних ігор ринок мобільних ігор є вельми перспективним, оскільки велика кількість користувачів готова витрачати гроші на гру. У майбутньому, ймовірно, з'являться нові технології, які дозволять розробникам гіпер-казуальних ігор створювати ще більш захоплюючі інтерактивні ігри з покращеною графікою та звуком. Також можна очікувати збільшення кількості рекламодавців, які зацікавлені в співпраці з розробниками ігор.

РОЗДІЛ 2. ВИБІР ЗАСОБІВ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

2.1 Ігровий двигун

Ігровий двигун - це програмне забезпечення, яке використовується для створення відеоігор. Він містить в собі набір інструментів та функцій для розробки, дизайну та реалізації геймплею, а також рендерингу графіки та обробки звуку.

Ігровий двигун можна розглядати як засіб забезпечення взаємодії між різними елементами відеоігри, він забезпечує швидкий та ефективний процес розробки гри. Більшість розробників відеоігор використовують готові ігрові двигуни, такі як Unreal Engine або Unity, оскільки це дозволяє їм зосередитися на розробці геймплею та інших аспектах гри.

Ігровий двигун має різні компоненти, такі як рушій фізики, система анімації, система штучного інтелекту, система візуалізації та рендерингу, система звуку, система керування та інші. Кожна з цих систем може бути налаштована та відрегульована розробником відповідно до вимог конкретної гри.

Ігровий двигун може бути використаний для створення різних жанрів відеоігор, від екшенів та шутерів до головоломок та інших категорій. Завдяки розвитку технологій, ігрові двигуни стають все більш потужними та вдосконалюються з кожним роком.

Ігровий двигун є надзвичайно важливим елементом при створенні відеоігор. Він забезпечує швидкий та ефективний процес розробки гри та дозволяє розробникам зосередитися на творчому процесі розробки геймплею та інших аспектів гри. Також ігровий двигун відповідає за роботу графіки, звуку, штучного інтелекту та багатьох інших аспектів, що роблять гру повноцінною.

Загалом, ігровий двигун - це ключовий компонент у створенні якісних та успішних ігор. Він дозволяє розробникам зосередитися на головній меті - створенні цікавої та захоплюючої гри, забезпечуючи оптимальну роботу всіх її складових.

2.2 Компоненти ігрового двигуна

Ігровий двигун є основою для розробки ігор і включає в себе різноманітні компоненти, які співпрацюють між собою для створення ігрового середовища. Тому дослідимо основні компоненти ігрового двигуна.

2.2.1 Ввід

Цей компонент відповідає за збір введення від користувача і передачу його до інших компонентів двигуна. Це може включати в себе обробку натискань на клавіші, рухи миші, торкання дисплея на мобільних пристроях та інші дії, що виконуються користувачем.

Ввід дуже важливий для ігрового досвіду, оскільки він дозволяє користувачам взаємодіяти з грою та відчувати контроль над героями та ігровими об'єктами. Поганий ввід може зруйнувати досвід гравців та зробити гру неприємною для гри.

Для оптимальної роботи ввід повинен бути максимально точним та надійним. Це означає, що гра повинна бути розроблена з урахуванням можливих різних введень та повинна відповідати на них належним чином. Наприклад, якщо гравець натискає клавішу стрілки вгору, персонаж гри повинен рухатися вгору.

Крім того, ввід може бути налаштований гравцем в залежності від їхніх вподобань та потреб. Гравці можуть вибрати різні схеми керування, такі як клавіатура та миша на ПК, екран сенсорного дисплея на мобільних пристроях, геймпади для консолей та інші пристрої. Тому розробники повинні забезпечити належну підтримку різних типів вводу, щоб користувачі могли насолоджуватися грою в тому форматі, який їм найбільше подобається.

2.2.2 Графіка та графічне API

Цей компонен відповідає за візуальну складову гри, включаючи зображення, текстури, анімацію та ефекти. Графіка забезпечує гравцеві візуальний зв'язок з грою, а тому є одним з ключових аспектів геймплею.

Графічне API - це програмне забезпечення, що дозволяє ігровому двигуну взаємодіяти з графічним обладнанням комп'ютера або пристрою. Це дає розробникам можливість створювати складну та реалістичну графіку, використовуючи технології, які прискорюють процес рендерингу. Графічне API також дозволяє розробникам контролювати відображення графіки на різних платформах та пристроях.

У більшості сучасних ігрових двигунів використовуються графічні API, такі як OpenGL, DirectX та Vulkan. Ці API дозволяють розробникам створювати складну 2D та 3D графіку, включаючи освітлення, тіні, текстури та частинки. Крім того, графічні API можуть бути оптимізовані для різних пристроїв та платформ, що дозволяє досягати максимальної продуктивності та ефективності.

Нарешті, графічна система взаємодіє з іншими компонентами ігрового двигуна, такими як фізика та аудіо, щоб створити повноцінний ігровий досвід. Кращі ігрові двигуни забезпечують високоякісну графіку, що дозволяє розробникам створювати неймовірні світи та персонажів, які допомагають зануритися гравця в гру.

Нарешті, графічна система взаємодіє з іншими компонентами ігрового двигуна, такими як фізика та аудіо, щоб створити повноцінний ігровий досвід. Кращі ігрові двигуни забезпечують високоякісну графіку, що дозволяє розробникам створювати неймовірні світи та персонажів, які допомагають зануритися гравця в гру.

2.2.3 Штучний інтелект

Штучний інтелект - це система, яка дозволяє ігровим об'єктам розуміти оточуючий світ, приймати рішення та взаємодіяти з гравцем і іншими об'єктами в грі. Штучний інтелект дозволяє розробникам створювати більш складні та цікаві ігри, де ігрові об'єкти можуть реагувати на дії гравця та навіть взаємодіяти між собою.

Одним з основних елементів штучного інтелекту є алгоритми прийняття рішень. Ці алгоритми дають можливість ігровим об'єктам приймати рішення на основі отриманої інформації про гравця та оточуючий світ. Різні алгоритми прийняття рішень можуть використовуватися в залежності від типу ігри та задачі, яку має виконати ігровий об'єкт.

Інший важливий компонент штучного інтелекту - це системи поведінки, які відповідають за взаємодію між ігровими об'єктами. Системи поведінки визначають, як ігрові об'єкти повинні реагувати на дії гравця та інші події в грі. Вони дають можливість створювати складні сценарії, де ігрові об'єкти можуть виконувати різні дії, залежно від ситуації.

Ще один важливий елемент штучного інтелекту - це системи взаємодії між ігровими об'єктами. Ці системи дають можливість ігровим об'єктам взаємодіяти між собою та з гравцем. Вони дозволяють створювати складні сценарії гри, де поведінка персонажів залежить від різних факторів, таких як дії гравця, стан середовища та інших персонажів. Штучний інтелект може бути використаний для створення різних типів поведінки персонажів, таких як прості манекени, що діють згідно з певними правилами, або складні персонажі з унікальною поведінкою та взаємодією з навколишнім світом.

Один з основних використань штучного інтелекту в іграх - це створення ворогів, які мають інтелект на рівні гравця. Вони можуть взаємодіяти з ним, розуміти стратегію гравця та намагатися уникати або знищувати його. Крім того, штучний інтелект може бути використаний для створення персонажів-партнерів, які допомагають гравцеві в його подорожі.

Також за допомогою штучного інтелекту можна створювати реалістичні ефектів у грі, таких як поведінка води, рух хмар чи створення різних ефектів освітлення. Це дозволяє створювати дуже реалістичні та живі ігрові світи, які сприймаються гравцями як живі організми.

Штучний інтелект також може бути використаний для оптимізації гри, зменшення навантаження на процесор та покращення продуктивності. Наприклад,

він може розподіляти обчислення на різні потоки та процеси, або зменшувати кількість деталей в грі на слабших пристроях для покращення швидкості роботи.

2.2.4 Звукова система

Дозволяє забезпечити реалістичне відтворення звуків, музики та інших аудіо ефектів в ігровому процесі. Звукова система також має важливу роль у створенні належної атмосфери в грі, допомагаючи гравцю краще відчути та сприйняти події на екрані.

Одним з основних компонентів звукової системи є мікшер, що дозволяє змішувати різні звукові джерела та відтворювати їх одночасно. За допомогою мікшера можна відтворювати звукові ефекти в залежності від різних параметрів, наприклад, відстані до звукового джерела, часу затримки та інших факторів.

Іншим важливим компонентом звукової системи є генератор звуків, що дозволяє створювати різноманітні звукові ефекти та музику. Генератор звуків може використовувати різні типи синтезу звуку, такі як аналоговий, фізичний або семпловий синтез.

Окрім цього, звукова система може включати різні звукові ефекти, такі як ехо, реверберація, кімнатні ефекти та інші. Ці ефекти можуть використовуватися для покращення якості звукового відтворення та створення більш реалістичної атмосфери в грі.

Звукова система повинна може мати інтерфейс для роботи зі звуковими файлами та редагування звукових ефектів. Це дозволяє розробникам ігор створювати та редагувати звукові ефекти безпосередньо в інтегрованому звуковому редакторі. Зазвичай він має наступні функції: налаштування гучності, зміна темпу, панорамування, еквалайзер, ефекти реверберації та ехо, а також можливість створення музичних композицій і редагування їхньої структури.

Звукова система також має важливе значення для геймплею. Звукові ефекти та музика допомагають створювати атмосферу гри та передавати емоції гравцеві. Наприклад, звук кроків може допомогти гравцеві зрозуміти, що його персонаж

рухається, а звук вибуху може передати відчуття небезпеки. Крім того, музика в грі може відображати настрій та емоції персонажів, створювати певну атмосферу та додавати драматизму до сюжету.

2.2.5 Мережеві комунікації

Ігрові застосунки можуть мати різні форми мультиплеєру, включаючи локальний мультиплеєр, мультиплеєр через Інтернет та мережевий мультиплеєр. Для того, щоб забезпечити гладку гру в режимі мультиплеєру, ігровий двигун повинен мати можливість швидко передавати дані між гравцями та серверами.

Мережеві комунікації можуть бути реалізовані за допомогою різних протоколів, таких як TCP (Transmission Control Protocol) або UDP (User Datagram Protocol). TCP забезпечує надійну передачу даних, але при цьому затримує їх, що може спричинити проблеми з часом відгуку. UDP, навпаки, не забезпечує надійну передачу даних, але при цьому дозволяє передавати їх швидко, що особливо важливо для онлайн-ігор.

Іншим важливим аспектом мережевої комунікації є безпека. Ігровий двигун повинен забезпечити захист від хакерських атак та підробки даних. Це може бути досягнуто за допомогою шифрування даних та автентифікації користувачів.

Крім того, ігровий двигун повинен мати можливість масштабуватись для підтримки більшої кількості гравців та серверів. Це може включати в себе використання хмарних сервісів, розподілену обробку даних та інші технології.

Загалом, мережева комунікація є важливим компонентом ігрового двигуна, оскільки вона дозволяє гравцям знаходитись у зв'язку один з одним та забезпечує гладку гру в режимі мультиплеєру.

Крім синхронізації, мережева комунікація також забезпечує можливість передачі різних даних між гравцями, таких як стан гравця, його позиція, обладнання та інші атрибути. Ці дані можуть використовуватися для створення різноманітних ігрових ефектів, таких як співпраця між гравцями, битви за територію, рейди тощо.

2.3 Аналіз популярних ігрових двигунів

Серед найпопулярніших ігрових двигунів у світі можна винести Unreal Engine, Unity та Godot. Проаналізуємо їх більш детально.

2.3.1 Unreal Engine

Unreal Engine - це один з найпопулярніших ігрових двигунів в світі, розроблений компанією Epic Games. Він використовується для створення ігор на різних платформах, включаючи ПК, консолі та мобільні пристрої. Unreal Engine пропонує широкий спектр функцій, таких як потужний редактор ігрових рівнів, гнучкий інструментарій штучного інтелекту, інтуїтивний інтерфейс користувача та потужна звукова система.

Однією з найбільш відомих особливостей Unreal Engine є його графічний двигун. Він забезпечує високу якість графіки, включаючи підтримку різних ефектів освітлення та тіней, фотореалістичну обробку зображень, та багато іншого. Unreal Engine також має велику спільноту розробників, яка активно розвиває та доповнює двигун, надаючи користувачам безкоштовний доступ до різноманітних плагінів та інструментів.

Unreal Engine також має великий набір вбудованих інструментів, які допомагають створювати геймплей та контент, такі як інструментарій з штучного інтелекту та фізичний двигун. Unreal Engine також підтримує різні платформи, включаючи ПК, консолі Xbox та PlayStation, а також мобільні пристрої.

Unreal Engine пропонує широкі можливості для розробників, які працюють в команді. Розробники можуть спільно працювати над проектом, ділитися ресурсами та спільно використовувати інструменти. Unreal Engine також має вбудовані засоби для тестування, що дозволяє розробникам виявляти та виправляти помилки, що допомагає створюват ь деталізовані ігрові світи. Unreal Engine має велику кількість інструментів для розробки високоякісних графічних об’єктів, включаючи високоякісну фізику, освітлення, тіні та реалістичні матеріали. Крім того, двигун

має потужну систему штучного інтелекту, що дозволяє розробникам створювати складні поведінкові сценарії для персонажів та ігрових об’єктів.

Unreal Engine також має досить велику спільноту розробників, яка постійно вдосконалює та розширює функціонал двигуна. До того ж, Unreal Engine надає розробникам доступ до відкритих джерел, що дозволяє вносити зміни в ісходний код двигуна та створювати власні плагіни та розширення.

Однак, Unreal Engine вимагає від розробників певного рівня кваліфікації та досвіду в програмуванні. Крім того, Unreal Engine є комерційним продуктом, що означає, що розробники мають платити за його використання та випуск готових ігор на ринок.

У даний час Unreal Engine використовується для створення високоякісних та популярних ігор, таких як Fortnite, Gears of War, Unreal Tournament та інших. Двигун також використовується для створення віртуальної реальності та інтерактивних візуалізацій у таких галузях, як архітектура, медицина та інженерія.

Плюси:

- Велика та активна спільнота розробників та користувачів Unreal Engine.
- Великий набір інструментів для створення ігор різних жанрів та платформ.
- Кроспатформенність.
- Потужна графічна система.

Мінуси:

- Не простий в освоєнні.
- Досить важким для маленьких проектів.
- Не дуже підходить для 2D-ігор, оскільки більше орієнтований на 3D проєкти.
- Вимагає певних ресурсів комп'ютера, щоб працювати належним чином.

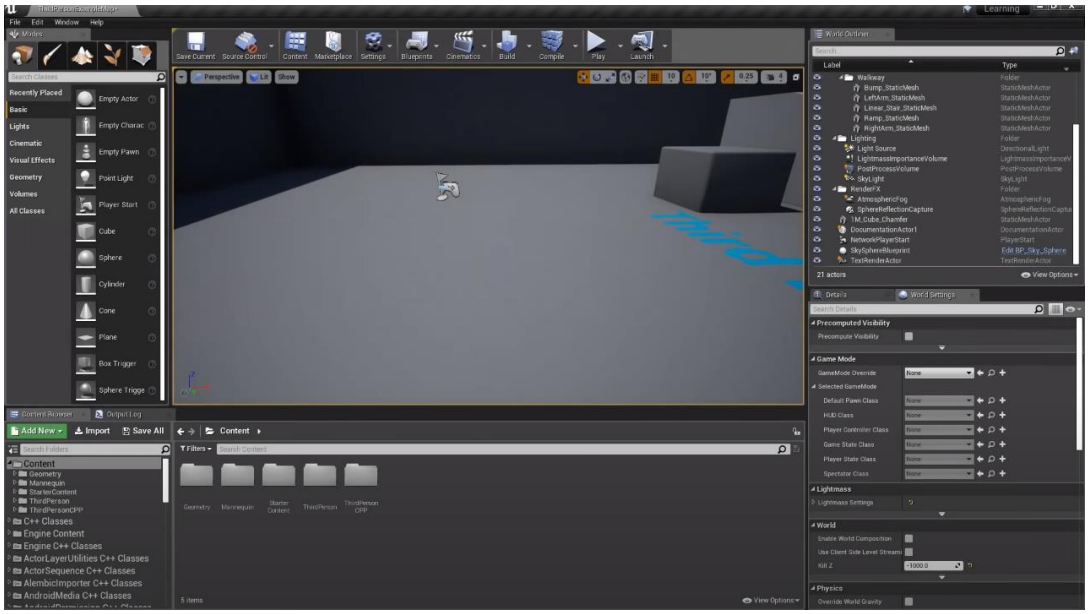


Рисунок 2.1 - Демонстрація інтерфейсу Unreal Engine

2.3.2 Unity

Unity - це ігровий двигун, що був створений у 2005 році компанією Unity Technologies. Його головна перевага полягає в тому, що він дозволяє розробляти ігри для різних платформ, таких як Windows, MacOS, Android, iOS, Xbox, PlayStation і т.д. Це забезпечує широку аудиторію та можливість розробки мобільних ігор. Крім того, Unity має велику спільноту розробників, які надають різноманітні ресурси для навчання та розробки.

Одна з істотних особливостей Unity - це його візуальний редактор, що легко дозволяє створювати, редагувати та налаштовувати об'єкти та сцени в ігрі. Unity також має багато компонентів, які дозволяють додавати різноманітні функції до ігри, такі як графіка, звук, фізика, штучний інтелект і т.д.

Unity пропонує широкий набір інструментів для створення ігор, зокрема графічний редактор, інструменти роботи з фізикою, засоби роботи зі штучним інтелектом та багато іншого. Основною мовою програмування, яку використовують для скриптування в Unity, є C#.

Unity має досить велику кількість готових модулів та інтегрованих сервісів, які дозволяють розробникам створювати багатофункціональні ігри. Один з найважливіших сервісів Unity - це магазин активів, де розробники можуть знайти

безліч безкоштовних та платних готових моделей, текстур, звуків та іншого контенту, який допоможе прискорити розробку ігор.

Плюси:

- Можливість створення ігор для різних платформ.
- Велика спільнота користувачів.
- Швидкість розробки.
- Багата функціональність.

Мінуси:

- Може мати проблеми з продуктивністю на старих та менш потужних пристроях.
- Обмеження на безкоштовній версії.
- Може мати проблеми зі сумісністю з деякими платформами.

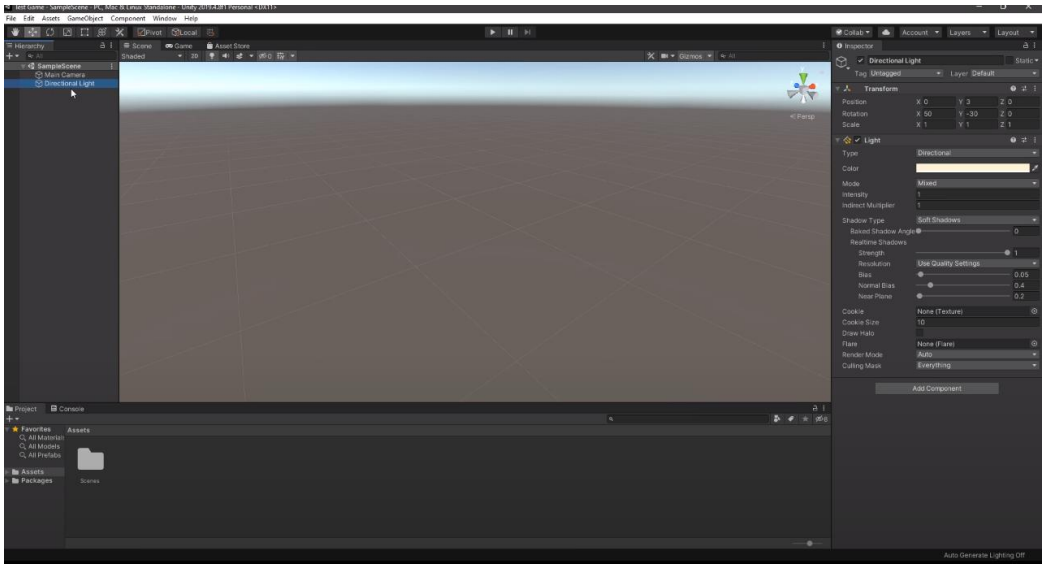


Рисунок 2.2 - Демонстрація інтерфейсу Unity

2.3.3 Godot

Godot - це безкоштовний відкритий ігровий двигун з візуальним редактором, який відмінно підходить для розробки 2D і 3D ігор. Він був створений в 2014 році і має розширювану архітектуру, що дозволяє додавати новий функціонал. Godot підтримує розробку для різних платформ, таких як Windows, macOS, Linux, Android та iOS.

Godot має вбудовану систему фізики, що дозволяє розробникам легко створювати реалістичну фізику для своїх ігор. Двигун також має вбудовану систему частинок, яка дозволяє створювати різні ефекти, такі як вогонь, дим та вода. Godot також має вбудовану систему штучного інтелекту, яка дозволяє розробникам створювати різні типи поведінки для персонажів.

Одним з головних переваг Godot є візуальний редактор, який дозволяє створювати ігри без написання коду. Редактор має вбудовану бібліотеку з компонентами, які можна додавати до сцен, що дозволяє швидко створювати прототипи ігор. У разі потреби розробники можуть звернутися до письмового коду, щоб додати більше функціональності.

Ще однією перевагою Godot є спільнота розробників, яка зростає з кожним роком. У спільноті є безліч корисних ресурсів, таких як документація, уроки та форуми, де розробники можуть отримати підтримку та поради від інших членів спільноти.

Узагалі, Godot - це потужний ігровий двигун, який може задовольнити потреби як початківців, так і досвідчених розробників. Він є ідеальним вибордля тих, хто шукає простоту та ефективність.

Один з головних плюсів Godot - це його відкритий вихідний код та безкоштовність. Розробники можуть отримати доступ до повного набору функцій безкоштовно, не потрібно платити за ліцензії або підписки. Це дозволяє розробникам зосередитися на творчості, а не на витратах.

Godot також має досить широкий набір функцій для розробки ігор, включаючи підтримку 2D та 3D графіки, фізичних ефектів, анімації та штучного інтелекту. Інтерфейс користувача є досить простим у використанні та дозволяє розробникам швидко створювати та тестувати свої ігри.

Плюси:

- Відкритий вихідний код.
- Підтримує різні платформи.
- Простий та легкий для вивчення інтерфейс.

- Інтегрований редактор графіки.
- Підтримує як 2D, так і 3D графіку.

Мінуси:

- Спільнота розробників досить невелика.
- Кількість навчальних ресурсів обмежена.
- Менш розширений інтерфейс користувача порівняно з іншими двигунами.

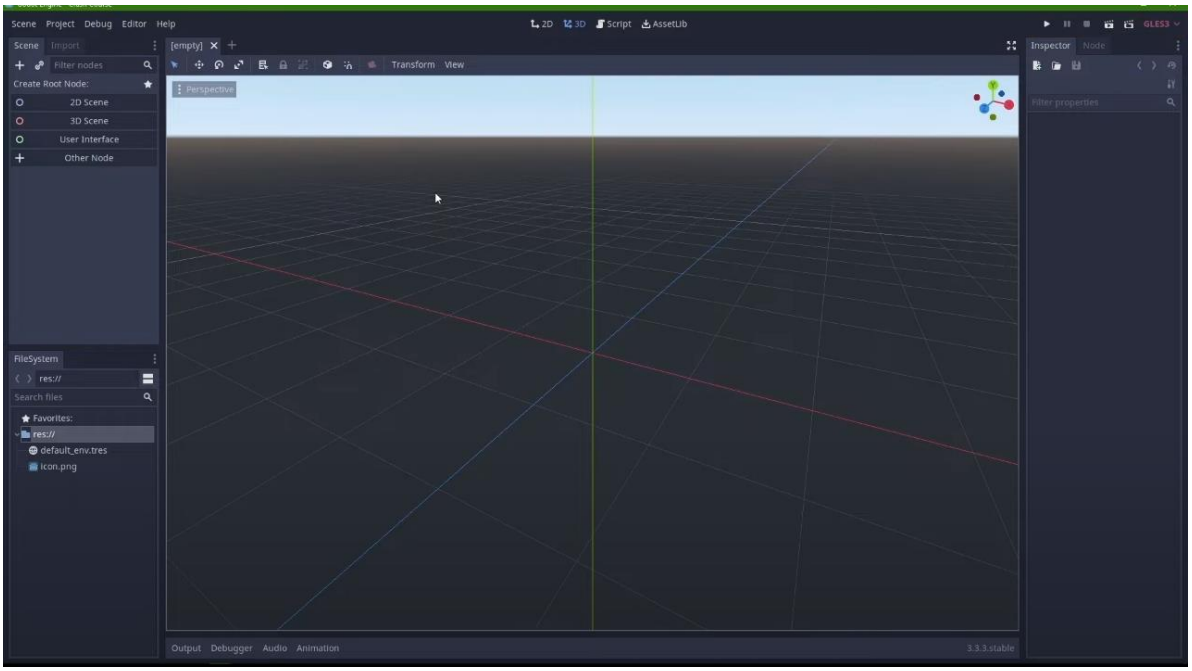


Рисунок 2.3 - Демонстрація інтерфейсу Godot

2.4 Аналіз середовищ розробки

Серед найбільш популярних та підходящих середовищ розробки для дипломного проєкту можна виділити Visual Studio та Rider.

2.4.1 Visual Studio

Visual Studio - це розроблене компанією Microsoft інтегроване середовище розробки (IDE), спеціально призначене для створення програмного забезпечення на платформі Windows, включаючи ігри. Цей потужний інструмент надає

розробникам ігор ряд переваг, дозволяючи їм ефективно створювати, тестувати, налагоджувати та впроваджувати ігрові проекти.

Однією з переваг Visual Studio є його потужність та гнучкість. Воно підтримує багато мов програмування, таких як C++, C#, Visual Basic та інші. Крім того, Visual Studio надає широкі можливості роботи зі зв'язками, розподіленою розробкою, тестуванням, профілюванням та іншими функціями, які дозволяють розробникам ефективно працювати над своїми проектами.

Іншою перевагою Visual Studio є його інтеграція з платформою Windows. Розробники можуть створювати ігри, що працюють на платформі Windows, використовуючи бібліотеки та інструменти, які надає це середовище розробки. Крім того, Visual Studio надає широкі можливості для розробки ігор на платформі Xbox, включаючи інструменти для розробки гри для консолі Xbox.

Також серед переваг Visual Studio є його спільнота розробників. Є велика кількість ресурсів, які розробники можуть використовувати для навчання та розвитку. Це включає в себе книги, відеоуроки, блоги та форуми, де розробники можуть отримати поради та підтримку від інших членів спільноти.

Проте, Visual Studio має й деякі недоліки. Один з них - це вартість. Повна версія Visual Studio може бути дуже дорога для початківців або невеликих команд. Крім того, Visual Studio вимагає потужного обладнання, щоб працювати на повну потужність. Це може бути зручно для досвідчених розробників, які бажають максимально оптимізувати свою роботу, але для новачків це може стати перешкодою у процесі вивчення. Однак, Visual Studio надає велику кількість інструментів для розвитку навичок програмування.

Окрім того, Visual Studio має вбудовану підтримку для багатьох мов програмування, включаючи C++, C#, Visual Basic, Python та інші. Це дозволяє розробникам працювати з тією мовою, яка їм найбільше підходить або з якою вони вже знайомі.

Ще один плюс Visual Studio - це підтримка засобів для розробки програмного забезпечення на різних платформах, таких як Windows, Linux, Android та інші. Це

дозволяє розробникам створювати програми для різних платформ з одного середовища розробки, що зменшує час та зусилля, витрачені на розробку.

Однак, недоліком Visual Studio може стати висока ціна та важкість у використанні для початківців. Також, для повноцінного використання Visual Studio необхідні досить потужний комп'ютер та відповідна конфігурація.

Усе ж таки, Visual Studio є потужним та зручним середовищем розробки, яке надає розробникам широкі можливості для створення програмного забезпечення на різних мовах програмування та платформах. Це може стати дуже корисним як для початківців, так і для досвідчених розробників, які шукають зручне та ефективне середовище для роботи.

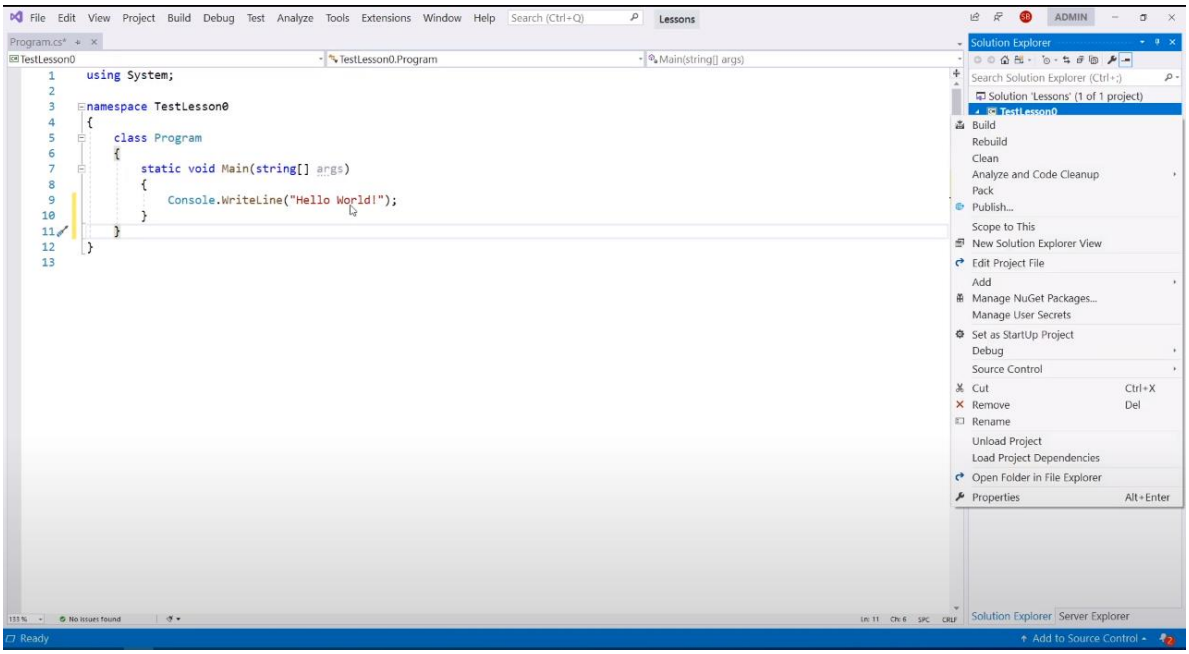


Рисунок 2.4 - Демонстрація інтерфейсу Visual Studio

2.4.2 Rider

Rider - це інтегроване середовище розробки (IDE), призначене для розробки програмного забезпечення на мовах C#, F# та VB.NET. Rider створений компанією JetBrains, яка відома своїми інструментами розробки, такими як IntelliJ IDEA для Java, PyCharm для Python та RubyMine для Ruby. Rider побудований на основі IntelliJ IDEA, але з налаштуваннями, що відповідають розробці програм на .NET.

Одним з головних переваг Rider є його інтеграція з іншими інструментами розробки програмного забезпечення від JetBrains, зокрема з ReSharper, WebStorm, PyCharm та іншими. Це дозволяє розробникам використовувати різні інструменти в одному середовищі, що спрощує розробку.

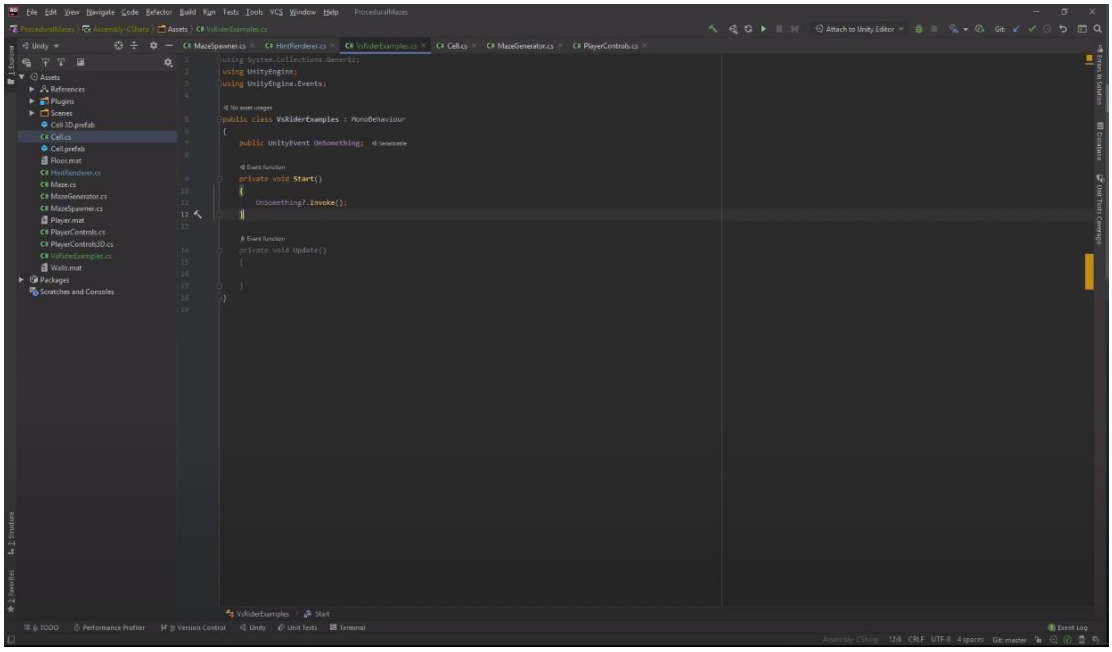


Рисунок 2.5 - Демонстрація інтерфейсу Rider

Серед недоліків Rider можна виділити платну ліцензію на комерційне використання та високі вимоги до обчислювальних ресурсів комп'ютера, на якому він використовується.

У загальному, середовище розробки Rider є потужним інструментом для розробки програмного забезпечення на мові C# та інших мовах програмування. Воно дозволяє розробникам працювати з різними інструментами в одному середовищі, що спрощує розробку, та підтримує роботу з різними платформами і системами контролю версій.

2.5 Висновки до розділу

У даному розділі було досліджено ігрові двигуни та їх компоненти, а також проведений аналіз популярних ігрових двигунів та середовищ розробки.
Вивчення компонентів ігрових двигунів показало, що кожен з них відповідає за конкретні функції гри. Наприклад, компонент вводу відповідає за обробку даних з клавіатури та миші, а компонент мережевих комунікацій - за забезпечення гри в режимі мультиплеєра.

Аналіз популярних ігрових двигунів, таких як Unreal Engine, Unity та Godot, показав, що кожен з них має свої переваги та недоліки. Unreal Engine є потужним та функціональним двигуном, з великою кількістю готових рішень та великим співтовариством розробників. Unity також є потужним та популярним, з великою кількістю ресурсів та плагінів, що сприяє швидкому та ефективному розробленню ігор. Тому саме Unity і був обраний для розробки проєкту. Godot є досить молодим двигуном, але має великий потенціал та можливості для розробки ігор різного рівня складності.

Також було проведено аналіз двох середовищ розробки - Visual Studio та Rider. Visual Studio має велику кількість функцій та плагінів, що робить його потужним інструментом для розробки будь-яких програм, включаючи ігри. Rider, з іншого боку, спеціалізується на розробці програм на мові програмування C# та має багато корисних функцій для розробки ігор, таких як дебагер, автодоповнення та інтеграція з різними ігровими двигунами.

РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РОЗРОБКА ГРИ

3.1 Опис геймплею гри

Рисунок 3.1 - Демонстрація гемплею гри

Гемплей гри полягає в тому, що гравець контролює машинку, яка рухається зі швидкістю вперед по нескінечній дорозі. Гравець має можливість переміщатися машинкою на праву або ліву смугу, щоб оминати перешкоди - інші машинки, які рухаються в зустрічному напрямку. Ці машинки рухаються зі своєю власною швидкістю, тому гравець повинен бути дуже уважним, щоб уникнути зіткнення.

Гравець може переміщати машинку в бік, перепригуючи на ліву або праву смугу, щоб оминути перешкоду. Щоб це зробити, гравець повинен свапнути екран у відповідну сторону, яка викличе стрибок машинки на іншу смугу. Також гравець може загальмувати на декілька секунд для цього йому потрібно затиснути лалець на екрані.

Гра пропонує гравцю різноманітні перешкоди, такі як інші машинки, дорожні знаки і т.д., що дозволяє робити гру більш складною і захопливою. Гравець може збирати монетки, які розкидані по дорозі, що дозволяє йому отримувати поінти за які можна купувати нові машинки або дорожні карти.

Ціль гри полягає в тому, щоб проїхати якомога далі по дорозі, збираючи якомога більше монеток і уникаючи зіткнення з перешкодами. Чим далі гравець проходить, тим складніше стає гра, і тим більше викликів він зустрічає на своєму шляху.

3.2 Реалізація гри

У рамках даного дипломного проекту була реалізована гра, в якій логіка контролюється за допомогою скриптів. Ці скрипти визначають поведінку різних об'єктів на ігровій сцені і в пам’яті. Було створено близько 90 файлів скриптів на мові C#, які розподілено на системи основні з яких описано в цьому розділі.

3.2.1 Система машинки гравця

Система машинки гравця складається з наступних скриптів (рис 3.2):

Death - відповідає за смерть гравця, якщо його машинка зіткнулася з іншими машинками або перешкодою.

ItemsCollector - відповідає за збір предметів на дорозі. Кожен зібраний предмет збільшує рахунок гравця.

MeterCounter - відповідає за підрахунок пройденого відстані. Він вимірює відстань, яку гравець пройшов на машинці.

TargetObjectCameraFollower - відповідає за камеру, яка слідкує за машинкою гравця. Камера буде пересуватися разом з машинкою гравця, щоб гравець завжди мав хороший огляд дороги.

VehicleEntity – це агрегатний клас який містить в собі деякі інші класи підсистем машинки і використовуется для ініцілізації їх а також служить як тип префабу для інстанціювання самоюї машинки при старті рівня.

VehicleLeader - Даний скрипт є компонентом гри для керування головним персонажем. Він містить в собі посилання на інші компоненти, такі як InputService для отримання вхідних даних від користувача і VehicleMover для руху машинки згідно з ввобом користувача.

VehicleMover - відповідає за рух машинки на дорозі. Цей скрипт визначає шлях руху машинки, керує її швидкістю та напрямком руху.

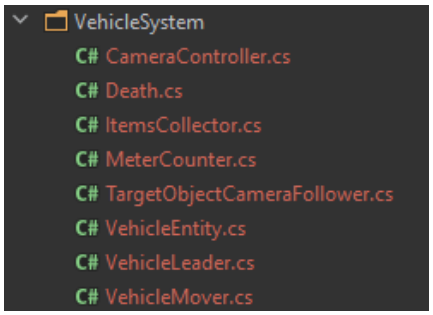


Рисунок 3.2 – Скрипти системи машинки гравця

3.2.2 Система машинки ворога

Система машинки ворога в грі складається з декількох скриптів (рис 3.3):

IEnemy – індифікуючий інтерфейс для ворогів.

EnemyCar - виконує роль ворожої машини у відеогрі. Він реалізує два інтерфейси: IPoolable та IEnemy. Інтерфейс IPoolable дозволяє перевикористовувати екземпляри класу, щоб зменшити використання ресурсів системи.

EnemyCarMover - відповідає за рух машини ворога. Він реалізує інтерфейс IPausable, що дозволяє зупиняти рух машинки при паузі гри.

EnemyCarType – перечислення доступних типів ворожих машинок.

ObstacleChecker – скрипт якій індифікує перешкоди у напрямку руху ворожої машинки і викидає подію у разі перешкоди.

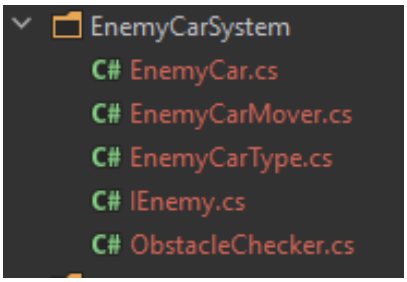


Рисунок 3.3 – Скрипти системи машинки ворога

3.2.3 Система генерації дороги

Система генерації дороги включає два скрипти: RoadView та RoadGenerator.

RoadView відповідає за відображення окремих ділянок дороги та її оточення. Клас приймає вхідний параметр типу MapType, що вказує, яку конкретну ділянку дороги необхідно відобразити. На основі цього параметру він зчитує зі зберігаючогося опису інформацію про матеріали та об'єкти, які повинні бути відображені, та налаштовує відповідні властивості MeshRenderer.

Крім того, клас додає на сцену об'єкт типу EnviromentLayerPrefab (рис 3.4), який містить об'єкти декору (такі як дерева, будинки тощо) для покращення візуальної привабливості дороги.

RoadGenerator відповідає за генерацію дороги. Він містить список Transforms, які відповідають за розташування дорожніх ділянок, та переміщує їх у залежності від руху гравця. Кожен раз, коли гравець досягає певної відстані від початку дороги, RoadGenerator переміщує першу ділянку на кінець списку і збільшує зміщення на фіксовану величину.

Крім того, RoadGenerator містить метод Initialize, який налаштовує RoadView для кожної ділянки дороги на початку гри. Метод приймає Transform гравця та тип дороги (MapType) як вхідні параметри.

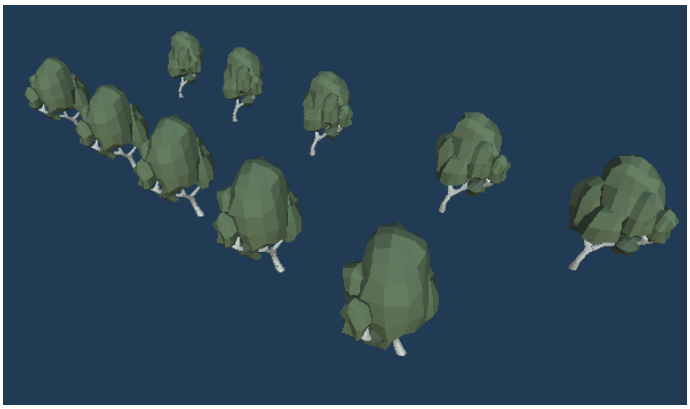


Рисунок 3.4 – EnviromentLayerPrefab одної із карт

Таким чином, система генерації дороги забезпечує плавний рух дороги та відображення відповідних матеріалів та об'єктів для кожної ділянки. Вона дозволяє створювати різні типи доріг, які можна змінювати залежно від рівня гри або установлених налаштувань.

3.2.4 Система спавну

Для того, щоб реалізувати спавн об'єктів в грі, потрібна система, яка відповідає за створення та розміщення їх на сцені. Система спавну (від англ. spawner system) - це один зі способів реалізації цієї задачі.

Система спавну представлена у вигляді абстрактного класу BaseSpawner та його реалізацій для кожного типу спавн об’єктів. Цей клас містить загальні поля та методи, які є необхідними для реалізації спавну об'єктів. Також, він має реалізації для кожного типу спавн об'єктів (машинки ворогів, поінти та перешкоди), які наслідуються від нього та додатково розширюють його функціональність.

Система спавну відповідає за створення та розміщення на сцені ігрових об'єктів з заданим інтервалом часу. Базовий клас BaseSpawner містить загальний функціонал для всіх реалізацій спавнерів. Він містить налаштування часових інтервалів спавну та деспавну, координати гравця, відстань між спавнером та гравцем, список активних об'єктів, координати спавну та методи для запуску/зупинки спавну та обробки його поведінки при паузі гри.

Кожна конкретна реалізація спавнера, наприклад одна з них, EnemyCarsSpawner, наслідується від базового класу BaseSpawner і реалізує метод SpawnLogic(), який визначає, який тип об'єктів потрібно створити, і які координати та орієнтацію задати. Крім того, в класах-спавнерах можна перевизначити метод ClearActiveObject(), який викликається при очищенні пулу об'єктів під час переходу на новий рівень або під час закінчення гри.

3.2.5 Система HUD інтерфейсу

Система HUD інтерфейсу (або просто HUD) - це частина інтерфейсу користувача, яка відображається на екрані гравця і містить інформацію про гру або стан персонажа. HUD може містити різні компоненти, такі як карта, життя головного персонажа, показники броні, інформація про зброю, лічильник очків, час і т.д.

Основна функція системи HUD - забезпечити гравцеві зручний доступ до важливої інформації про гру. Для цього, компоненти HUD можуть бути розміщені на екрані в різних місцях, залежно від того, яка інформація є найважливішою в даній ситуації гри. Також, система HUD може бути налаштована на різні способи, щоб відображати інформацію на екрані, такі як розмір, кольори, стиль, шрифти і т.д.

В грі, про яку йдеться, система HUD була реалізована за допомогою компонента Canvas в Unity. Canvas - це елемент інтерфейсу, який дозволяє створювати 2D-елементи, такі як кнопки, текстові поля та інші, та розміщувати їх на екрані.

Рисунок 3.5 – HUD з гемплею

HUD гемплею (рис. 3.5) містить показники проїханих метрів і зібраних поінтів, які оновлюються залежно від дій гравця. Це дозволяє гравцю відстежувати свій прогрес в грі та порівнювати його з попередніми результатами.

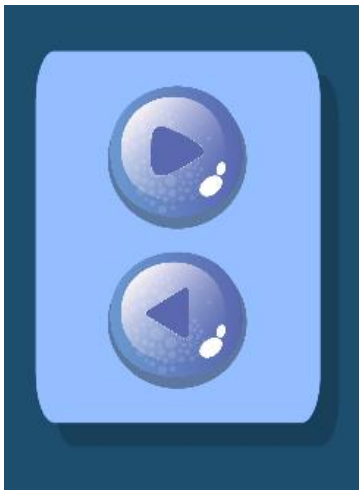


Рисунок 3.6 – інтерфейс віконця паузи

Також в системі HUD є кнопки паузи, які дозволяють гравцеві зупинити гру на деякий час, віконце паузи (рис. 3.6), яке відображається при натисканні кнопки паузи, та віконце завершення гри (рис. 3.7), яке з'являється після того, як гравець зіткнувся з ворожою машинкою або іншою перешкодою.

Рисунок 3.7 – інтерфейс віконця закінчення гри

3.2.6 Система паузи

Система паузи для гри представляє собою функцію, яка дозволяє гравцеві тимчасово зупинити гру, зберегти поточний стан гри та повернутися до неї пізніше. Зазвичай ця функція активується натисканням певної кнопки або комбінації клавіш на клавіатурі чи екрані.

Коли гравець активує функцію паузи, гра сповільнюється або повністю зупиняється, і на екрані з'являється меню паузи, де гравець може вибрати різні опції, такі як збереження гри, налаштування, вихід з гри та ін. Після того, як гравець вибере потрібну опцію та закриє меню паузи, гра продовжується з збереженого стану.

Система паузи в грі може бути корисною, коли гравцеві потрібно тимчасово покинути гру, але не хочеться починати її знову.

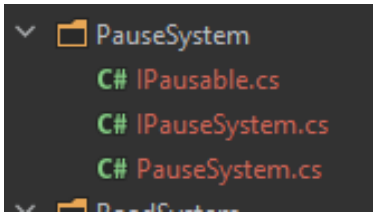


Рисунок 3.8 – Скрипти системи паузи

Дана система паузи (рис 3.8) представляє собою інтерфейс IPausable, який визначає метод SetPaused, що дозволяє встановлювати прапорець паузи для об'єктів, які його реалізують.

Також є інтерфейс IPauseSystem, який наслідується від IPausable та IService і визначає методи RegisterPausable та UnRegisterPausable для додавання та видалення паузабильних об'єктів зі списку, та метод SetPaused для встановлення прапорця паузи для всіх зареєстрованих об'єктів.

Клас PauseSystem реалізує інтерфейс IPauseSystem, та містить приватне поле _handlers, яке містить список паузабильних об'єктів. Також він містить публічне властивість IsPaused для отримання стану паузи, та реалізовує метод SetPaused,

який встановлює прапорець паузи та викликає метод SetPaused для всіх об'єктів, які були зареєстровані в _handlers.

Дана система дозволяє зупиняти роботу об'єктів, що її реалізують, через встановлення прапорця паузи з інших об'єктів за допомогою інтерфейсу IPauseSystem.

3.2.7 Система обжект пулінгу

Патерн Object Pooling є популярним підходом у розробці ігор, який дозволяє ефективно використовувати обмежені ресурси, такі як пам'ять та процесорний час, для створення гри з високою продуктивністю.

Патерн Object Pooling зменшує навантаження на ресурси шляхом створення пулу об'єктів, які можуть бути повторно використані замість того, щоб створювати нові об'єкти кожного разу. Коли об'єкт більше не потрібний, його можна повернути до пула, замість того, щоб його знищувати.

У даному проєкті патерн Object Pooling реалізований за допомогою трьох файлів: двох класів та інтерфейсу. Інтерфейс називається IPoolable і використовується для взаємодії з ігровими об'єктами, які зберігаються в пулі. Основним класом є ObjectPool. У грі присутні кілька видів об'єктів, які потрібно зберігати в пулі, тому клас має словник для цієї мети. Кожен вид ігрового об'єкту зберігається окремо в пулі, що сприяє зручній роботі з пулом. ObjectPool додає нові пункти в словник за потреби.

Клас ObjectPool також використовує патерн Singleton. Останнім класом, пов'язаним з пулом об'єктів, є PoolTask. У залежності від кількості видів ігрових об'єктів, що з'являються на сцені, може бути створено декілька екземплярів цього класу. PoolTask відповідає за додавання об'єктів в пул (або їх створення за потреби), зберігання їх у списку та вилучення об'єктів з пулу.

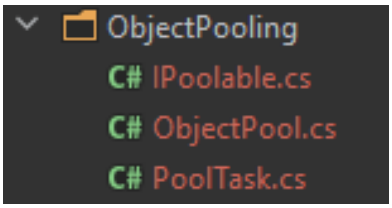


Рисунок 3.9 – Скрипти системи обжект пулінгу

3.2.8 Система головного меню

Система головного меню гри складається з кількох скрінів, кожен з яких має свою функціональність.

Головний екран (рис. 3.10) містить кнопки для доступу до інших скрінів: "Грати", "Налаштування", "Магазин" та "Рейтинг".

Рисунок 3.10 – Головне меню гри

При натисканні на кнопку "Налаштування", відкривається скрін налаштувань, де користувач може налаштувати гучність звуку та музики в грі.

Рисунок 3.11 – Налаштування

При натисканні на кнопку "Рейтинг", відкривається скрін з рейтингом гравця. Користувач може побачити список найкращих результатів.

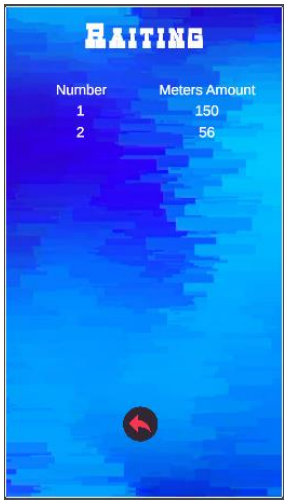


Рисунок 3.11 – Рейтинг

При натисканні на кнопку "Магазин", відкривається скрін магазину, де користувач може купувати нові машинки та карти за ігрові ітеми.

Рисунок 3.11 – Магазин

Усі скріни мають кнопку "Назад", яка дозволяє користувачу повернутися до головного екрану. Така система головного меню дозволяє користувачеві легко знаходити потрібні функції та налаштовувати гру на свій смак.

Що до реалізації кожен скрін має свій власний view скрипт, який відповідає за відображення контенту на екрані та методи для зміни його поведінки. Наприклад, view скрипт для головного скріну містить методи для відображення кнопок гри, магазину, налаштувань та рейтингу, а також методи для зміни відображення цих елементів в залежності від стану гейм стейту.

Для кожного view скрипту є контролер, який взаємодіє з гейм стейтом та забезпечує логіку взаємодії з головним станом гри. Наприклад, контролер для скріну налаштувань має методи для зміни гучності звуку та музики, які змінюють значення в головному стані гри.

Також є верхньорівневий контролер гри який містить зв'язки зі всіма view та контролерами, що забезпечує взаємодію між ними. Наприклад, при натисканні кнопки гри на головному скріні, головний контролер може викликати метод view скрипта для відображення скріну гри.

3.2.9 Система ігрового стейту

Ігровий стейт (game state) - це сукупність даних, які описують поточний стан гри. Ігровий стейт включає в себе всі елементи гри, такі як гравець, вороги, об'єкти на карті, стани анімацій та багато іншого.

Підхід до роботи з ігровим стейтом може різнитися в залежності від конкретної гри та її механік. Проте, зазвичай ігровий стейт може бути збережений у пам'яті або на диску в форматі файлу з певним розширенням.

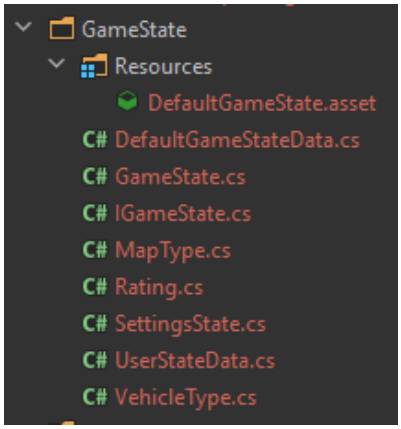


Рисунок 3.12 – Скрипти системи ігрового стейту

В даному проекті система стейту представлена двома дата об’єктами SettingsStateData і UserStateData, інтерфейсом IGameState та його реалізацією GameState (рис. 3.12).

Об'єкт SettingsStateData містить інформацію про звукові налаштування гри, зокрема про гучність музики та звуків. Об'єкт UserStateData містить інформацію про стан гри гравця, зокрема про те, чи зіграна гра, кількість набраних гравцем балів, поточний тип транспорту та тип карти, доступні типи транспорту та типи карт і рейтинг гравця.

Об’єкт GameState містить собі поля з дата об’єктами ігрового стейту та метод для зберезання стейту який зберігає стан стейту використовуючи ObjectPref описану низче.

Об'єкт GameState також містить константу зі стандартним іменем файлу для збереження дефолтного стану гри та ключі для збереження дата об'єктів в

PlayerPrefs. При створенні об'єкту GameState відбувається завантаження дефолтних дата об'єктів з файлу та ініціалізація ObjectPref з використанням цих дефолтних даних.

3.2.10 Система управління

Система управління представлена 5 скриптами (рис. 3.13): InputService, SwipeDetector, SwipeDirection, TouchDetector, TouchObserver

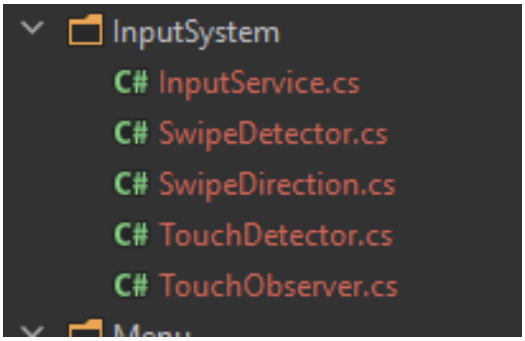


Рисунок 3.13 – Скрипти системи управління

InputService - це скрипт, який відповідає за обробку введення користувача. Він слідкує за натисканням користувача на екран та передає інформацію про ці дії в інші скрипти.

SwipeDetector - це скрипт, який виявляє рухи пальців на екрані сенсорного пристрою та передає інформацію про ці рухи в інші скрипти.

SwipeDirection - це перечислення, яке визначає напрямок руху пальця користувача на екрані. Цей перечислення використовується разом із SwipeDetector.

TouchDetector - це скрипт, який виявляє дотики користувача до екрану сенсорного пристрою та передає інформацію про ці дотики в інші скрипти.

TouchObserver - це скрипт, який слідкує за дотиками користувача до екрану та відслідковує їхні координати, стан та інші характеристики та має події для їх відслідкування.

3.2.11 Система Збереження

Unity має вбудований механізм для збереження та отримання даних в грі - PlayerPrefs. Вона є досить простою для використання та дозволяє зберігати налаштування, досягнення та інші потрібні дані між сеансами гри.

Одна з основних переваг системи збереження в PlayerPrefs - це її простота використання. Це означає, що розробникам не потрібно витрачати багато часу на налагодження складних механізмів збереження та відновлення даних гри.

Крім того, система збереження в PlayerPrefs Unity є дуже надійною та стійкою. Дані, збережені в PlayerPrefs, зберігаються на диску, тому вони не втратяться при випадковому вимкненні гри або системи. Це робить її ідеальним рішенням для збереження даних, які необхідно зберігати між різними сеансами гри.

Ще одна перевага системи збереження в PlayerPrefs - це швидкість доступу до даних. Оскільки дані зберігаються на диску, вони можуть бути легко відновлені при наступному запуску гри. Це робить їх доступними миттєво, що є важливим для гравців.

Система збереження в PlayerPrefs дозволяє зберігати дані різних типів, такі як числа, рядки та булеві значення. Це дозволяє розробникам зберігати будь-які дані, які необхідні для гри.

Так як Utity PlayerPrefs не працює з обьектами то для вирішення цієї проблеми було написано невеличку обгортку із 2 скриптів: Pref <T>, ObjectPref<T> ці скрипти дозволяють зберігати та отримувати об'єкти будь-якого класу у PlayerPrefs, який працює лише зі звичайними типами даних. Клас Pref<T> є абстрактним і містить загальну логіку для роботи з даними типу T. У цьому класі є методи для збереження, отримання та видалення значення за ключем. Крім того, в класі є властивість IsSaved, яка повертає true, якщо значення збережено в PlayerPrefs за вказаним ключем.

Клас ObjectPref<T> наслідується від Pref<T> і містить реалізацію методів Get() та Set() для зберігання та отримання об'єктів типу T в PlayerPrefs. Для збереження об'єкту в PlayerPrefs, об'єкт серіалізується у JSON-формат за допомогою бібліотеки Newtonsoft.Json. Для отримання об'єкту з PlayerPrefs, JSON-

строка розпаковується за допомогою тієї ж бібліотеки та десеріалізується у необхідний об'єкт класу T. Клас ObjectPref<T> обмежує тип T параметром where T : class, щоб забезпечити, що можна зберігати лише посилання на об'єкти (об'єкти, які можуть бути null).

3.2.12 Аудіо система

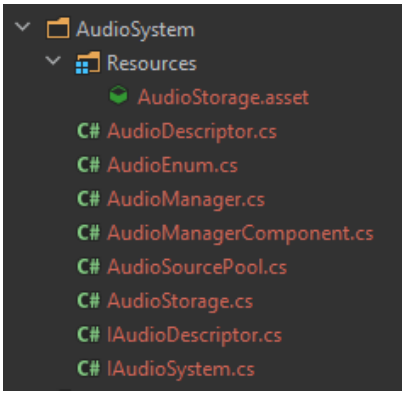


Рисунок 3.14 – Скрипти аудіо системи

Для більшої гнучкість в роботі зі звуком у грі, було вирішено написати свою власну аудіо систему обгортку над влаштованою Unity аудіо системою. Це може забезпечити більшу контроль над звуками і більшу комфортне використання із коду.

Ця аудіо система складається з двох основних класів: інтерфейсу IAudioSystem та його реалізації AudioManager. Інтерфейс IAudioSystem визначає методи, які можуть використовуватися в системі, а саме: PlayAudio, PlayBackground, StopBackground, SetAudioVolume і SetBackgroundVolume.

Клас AudioManager реалізує інтерфейс IAudioSystem і забезпечує конкретну реалізацію методів. Він містить посилання на AudioManagerComponent, який насправді забезпечує відтворення звуку. AudioManager передає виклики методів з інтерфейсу IAudioSystem до AudioManagerComponent.

Клас AudioManagerComponent є основним класом, який забезпечує відтворення звуку. Він містить різні поля, такі як AudioStorage, AudioSourcePool, _backgroundSource тощо, які допомагають зберігати і керувати джерелами звуку.

Основна логіка відтворення звуку реалізована за допомогою методу PlayingRoutine(). Цей метод використовує AudioSourcePool, щоб слідкувати за джерелами звуку, які в даний момент відтворюються. Якщо джерело звуку завершує свою роботу, то воно повертається до пула AudioSourcePool для подальшого використання.

Окрім того, AudioManagerComponent містить методи для відтворення звуку (PlayAudio та PlayBackground), зупинки відтворення (StopBackground), а також методи для налаштування гучності звуку (SetAudioVolume та SetBackgroundVolume).

3.2.13 Локатор сервісів

Паттерн Локатор Сервісів (від англ. Service Locator) - це шаблон проектування, який дозволяє знайти та отримати доступ до різних сервісів, які можуть бути доступні у додатку. Ідея полягає у тому, щоб мати централізований реєстр, який містить вказівники на всі доступні сервіси, і звертатися до них через цей реєстр.

У проекті на Unity паттерн Локатор Сервісів може бути використаний для спрощення доступу до різних сервісів, таких як звуковий двигун, система збереження даних, стан гри тощо.

В дипломному проєкті даний паттерн представлений двома сутностями: пустим інтерфейсом IService який відповідае за індифікацію сервісів і класс Locator у вигляді простого класу-одиночки з методами реєстрації, відміни реєстрації та отримання сервісу. Локатор дозволяє зареєструвати об'єкти-імплементації певних інтерфейсів і отримувати їх за потребою. В даному випадку, всі імплементації повинні реалізовувати інтерфейс IService, який відображає інтерфейс для всіх зареєстрованих сервісів.

3.2.14 Точки входу

Паттерн Entry Point (від англ. точка входу) - це структурний паттерн проектування, який дозволяє розміщувати основний код додатку в одному місці, яке відповідає за запуск програми і координування роботи всіх її компонентів.

У контексті Unity, Entry Point буде відповідати за запуск гри та ініціалізацію всіх необхідних систем для забезпечення їх коректної роботи.

Дипломний проект має три таких точки входу по одній на кожну ігрову сцену (рис 3.3). Сцена Entry існує тільки для відділення початкового ентру поінту який ініціалізує локатор сервісів для подальшого їх використання щоб не було конфліктів з порядком визовів колбек фукцій ві Unity. Entry Point від сцени Game містить методи, що викликаються при старті гри. В цьому скрипті описано, як ініціалізувати гравця, який керує транспортним засобом, а також різні ігрові об'єкти, які залежать від цього транспортного засобу, такі як вороги, предмети, перешкоди та генерація дороги. Також в цьому коді реалізовано функціонал туторіалу, який запускається при першому запуску гри. Entry Point від сцени Menu ініціалізуються об'єкти контролерів, керуючих різними елементами гри, такими як меню, магазини, рейтинг, налаштування тощо. При старті запускається ініціалізація деяких контролерів та налаштування звуків. При знизщенні об’єкту відбувається звільнення ресурсів, що використовуються в грі.

Рисунок 3.3 – Ігрові сцени

3.3 Тестування

Гру було протестовано за основним сценарієм такими як:

1. Відкрити гру
2. Вибрати машинку
3. Почати гру
4. Переміщувати машинку ліворуч або праворуч, щоб уникнути зіткнення з перешкодами
5. Збирати монетки на дорозі
6. Продовжувати гру, доки не стане неможливим уникнути зіткнення з перешкодами або не буде досягнуто максимальної відстані
7. Закінчити гру

Також було проведено тестування таких випадків:

- Перевірено, що гра запускається без помилок
- Перевірено, що гравець може вибрати машинку перед початком гри
- Перевірено, що гравець може переміщувати машинку на праву або ліву смугу, щоб уникнути зіткнення з перешкодами
- Перевірено, що гравець може збирати монетки на дорозі
- Перевірено, що гра закінчується, якщо гравець зіткнувся з перешкодою
- Перевірено, що гравець може купувати нові машинки та дорожні карти за зібрані монетки
- Перевірено, що гра продовжується після досягнення максимальної відстані, але стає більш складною
- Перевірено, що гра має функцію паузи та продовження гри.

Таблиця 3.1 - Тест кейси

Тест-кейс	Опис кейсу	Очікуваний результат	Фактичний результат	Відмітка
1	Запуск гри	Гра запускається без помилок	Працює згідно очікуваному результату	●
2	Рух вліво	Гравець може перемістити машинку на ліву смугу	Працює згідно очікуваному результату	●
3	Рух вправо	Гравець може перемістити машинку на праву смугу	Працює згідно очікуваному результату	●
4	Зіткнення з машинкою зі зворотного напрямку	Гравець може зіткнутися з машинкою, яка рухається в зворотному напрямку і програти	Працює згідно очікуваному результату	●
5	Збирання монеток	Гравець може збирати монетки, які розкидані по дорозі	Працює згідно очікуваному результату	●

Продовження таблиці 3.1 - Тест кейси

Тест-кейс	Опис кейсу	Очікуваний результат	Фактичний результат	Відмітка
6	Купівля нових машинок	Гравець може купувати нові машинки за зібрані поінти	Працює згідно очікуваному результату	●
7	Купівля нових дорожніх карт	Гравець може купувати нові дорожні карти за зібрані поінти	Працює згідно очікуваному результату	●

3.4 Висновки до розділу

У ході проектування та розробки гри було проведено дослідження геймплею та була зроблений його опис. Для реалізації гри було розроблено різні системи які розподілені за відповідальностями. Кожна з цих систем відіграє важливу роль у функціонуванні гри та дозволяє забезпечити користувачам більш зручне та комфортне використання гри.

Для забезпечення якості та надійності гри, було проведено тестування, яке дозволило виявити та виправити різноманітні помилки та недоліки. Під час тестування було використано тест-кейси для проведення систематичного та об'єктивного аналізу функціональності гри. Тестування дозволило підтвердити відповідність гри заданим вимогам, виявити та виправити помилки та недоліки, що забезпечило підвищення якості та надійності гри.

Таким чином, завдяки проведеному дослідженню геймплею, розробці та впровадженню систем реалізації гри, а також проведенню тестування та виправленню помилок та недоліків, було створено якісну та надійну гру, що забезпечує комфортну геймінговий досвід користувачам.

ВИСНОВКИ

У ході виконання даної дипломної роботи було розроблено мобільну гру "Traffic Jumble" у жанрі hyper casual для платформи Android з використанням ігрового рушія Unity та мови програмування C#. Гра пропонує користувачеві випробувати свою реакцію та навички управління шляхом маневрування автомобілем, уникання зіткнень з іншими транспортними засобами та збирання бонусів на шляху.

У процесі розробки гри було створено інтуїтивний ігровий інтерфейс, забезпечено плавну та реалістичну анімацію руху автомобіля, реалізовано систему взаємодії з об'єктами гри, а також реалізовано механізми генерації випадкових рівнів та підрахунку балів. Крім того, гра має зручне керування за допомогою сенсорного екрану мобільного пристрою, що дозволяє користувачеві легко контролювати рух автомобіля.

Результати тестування підтвердили, що гра працездатна і задовольняє вимоги, зазначені у висновку. Користувачі можуть насолоджуватися геймплеєм, виконуючи різноманітні маневри та отримуючи задоволення від гри.

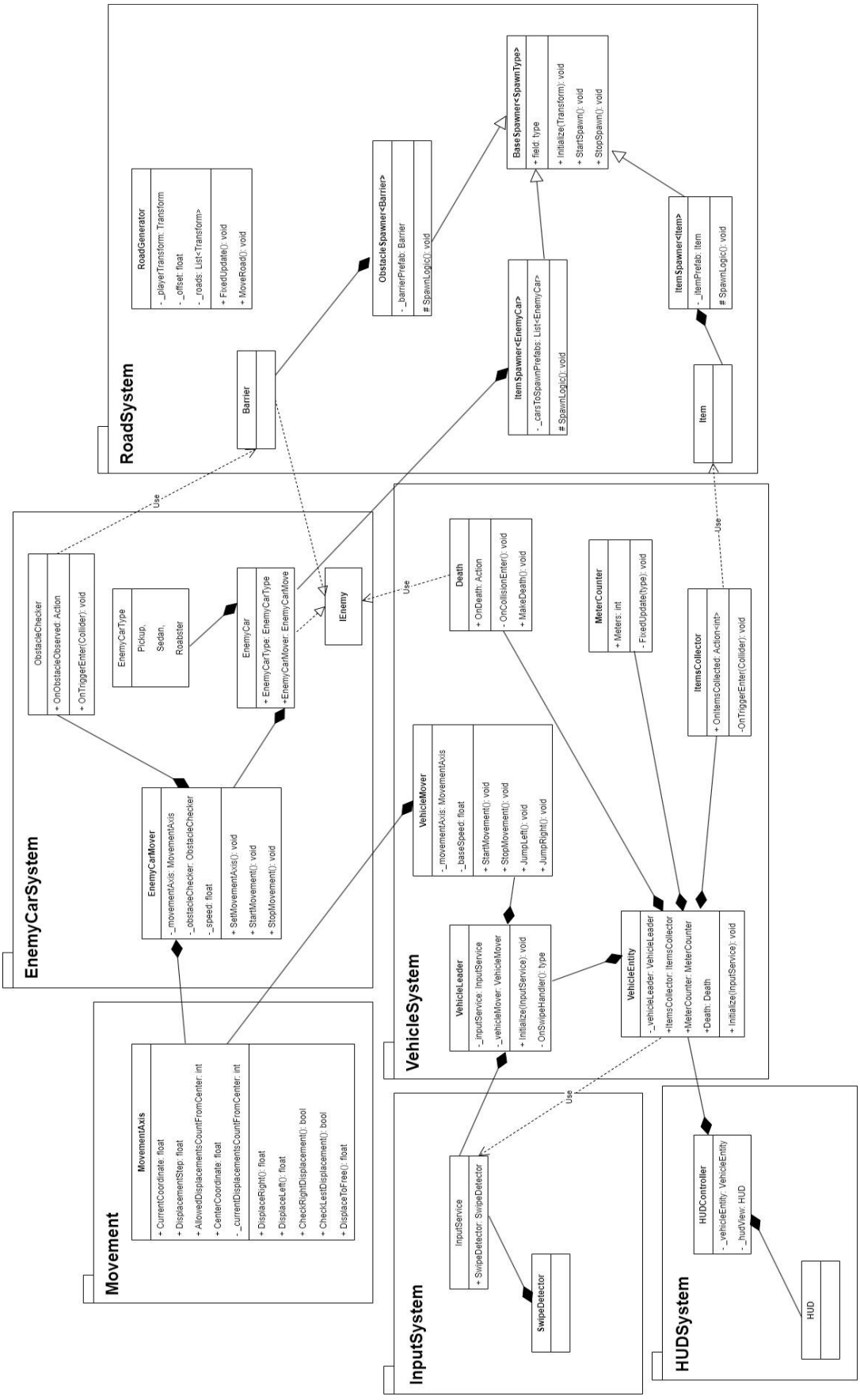
ПЕРЕЛІК ПОСИЛАНЬ

1. What are Video Games?. URL: https://www.twinkl.de/teaching-wiki/video-games
2. What Is Considered A Video Game? Definition And Examples | Game Design Lounge | Video Game Level, Character, and Story Design. *Game Design Lounge | Video Game Level, Character, and Story Design*. URL: https://gamedesignlounge.com/what-is-considered-a-video-game/
3. What are Gaming Consoles?. *Easy Tech Junkie*. URL: https://www.easytechjunkie.com/what-are-gaming-consoles.htm
4. Світовий ринок ігор у 2022 став дорожче усіх активів Ілона Маска. Підсумки року за версією Games Industry. *dev.ua*. URL: https://dev.ua/news/naipopuliarnishi-ihry
5. Mobile Gaming Market Trends and Growth Forecast Report, 2030. *P&S Intelligence*. URL: https://www.psmarketresearch.com/market-analysis/mobile-gaming-market
6. What is a Game Engine? | Studytonight. *Studytonight - Best place to Learn Coding Online*. URL: https://www.studytonight.com/3d-game-engineering-with-unity/game-engine
7. 8 Popular Mobile Game Genres. URL: https://www.ciit.edu.ph/mobile-game-genres-2019/
8. Mobile games: ranking by genre. *JeuMobi.com*. URL: https://www.jeumobi.com/en/genre-mobile-games/
9. Casual Mobile Games & Best Casual Games 2021 | ironSource. *ironSource*. URL: https://www.is.com/glossary/casual-games/
10. Hedger G. The Structure of Video Game Engines. *Medium*. URL: https://medium.com/@ghedger42/the-structure-of-video-game-engines-e29329f6ba2c

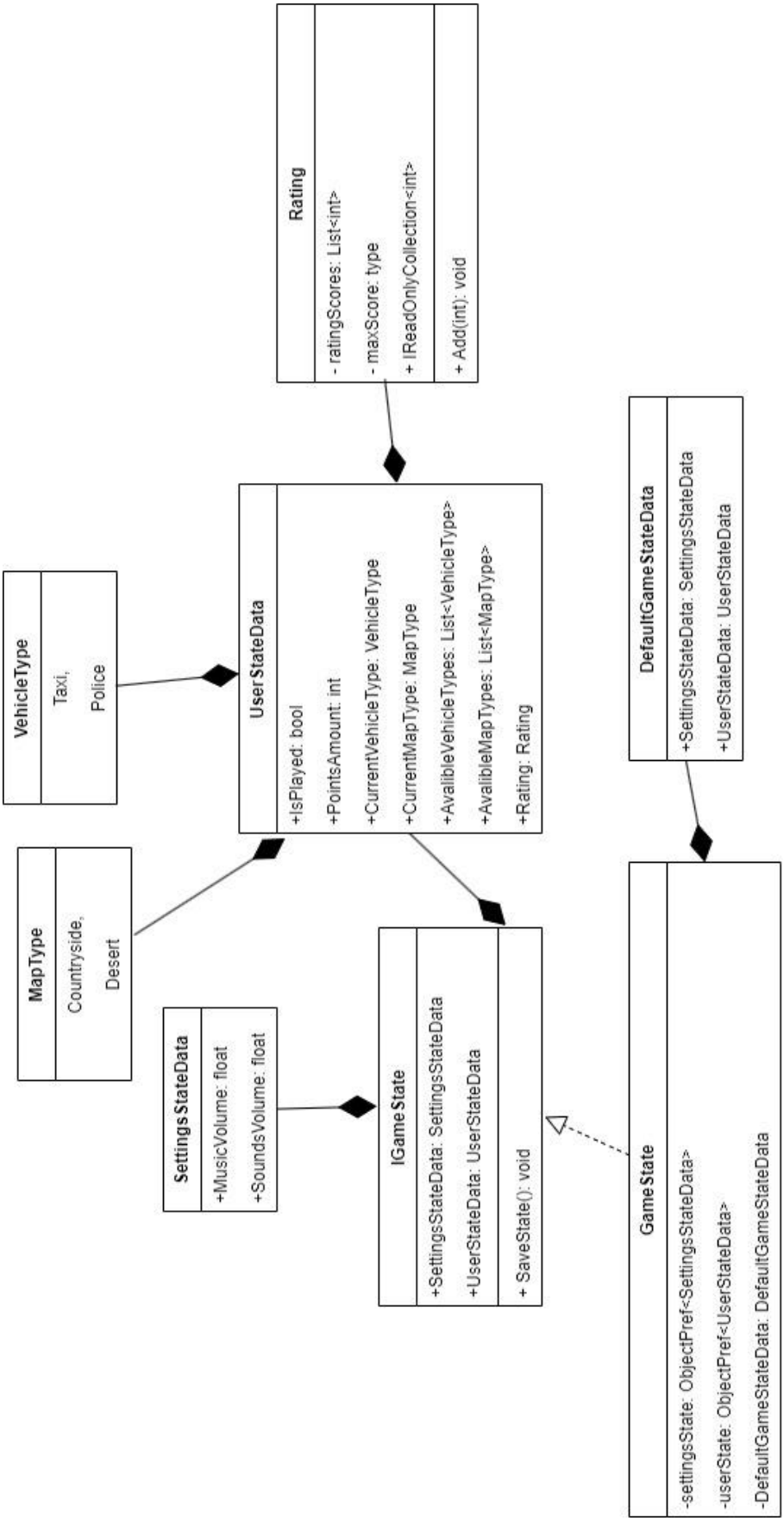
11. Introduction to Godot. *Godot Engine documentation*. URL: https://docs.godotengine.org/en/stable/getting_started/introduction/introduction_to_godot.html
12. What is Unity? Everything you need to know. *Android Authority*. URL: https://www.androidauthority.com/what-is-unity-1131558/
13. What Is Unreal Engine? - BairesDev Blog: Insights on Software Development & Tech Talent. *BairesDev Blog: Insights on Software Development & Tech Talent*. URL: https://www.bairesdev.com/blog/what-is-unreal-engine/
14. Visual Studio: IDE and Code Editor for Software Developers and Teams. *Visual Studio*. URL: https://visualstudio.microsoft.com/
15. Rider: The Cross-Platform .NET IDE from JetBrains. *JetBrains*. URL: https://www.jetbrains.com/rider/

Додаток А

Діаграма основних класів гемплею



Діаграма класів ігрового стореджу



Діаграма варіантів використання

Додаток В

Демонстраційні матеріали

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка гри "Traffic Jumble" у жанрі Hyper Casual з використанням ігрового рушія unity під мобільну платформу Android мовою C#

Виконав студент 4 курсу
групи ПД-42
Осьмак Владислав Романович
Керівник роботи
доцент кафедри ІПЗ Дібрівний Олесь Андрійович

Київ – 2023

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи – підвищення цікавості гемплею за рахунок використання гіперказуальних механік.

Об’єкт дослідження – ігровий процес у жанрі "hyper casual" під платформу Android.

Предмет дослідження – гра в жанрі "hyper casual" під платформу Android.

2

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Аналіз відеоігор і їх жанрів
2. Аналіз ринку гіперказуальних ігор для мобільних платформ.
3. Вивчення можливостей ігрового рушія Unity.
4. Проектування геймплею гри.
5. Пошук графіки та дизайну гри.
6. Проектування архітектури гри
7. Розробка гри.
8. Тестування гри та виявлення помилок.

3

АНАЛІЗ АНАЛОГІВ

Subway Surfers

Car Run 2

2D Car Runner

4

АНАЛІЗ АНАЛОГІВ

Показник	Subway Surfers	Car Run 2	2D Car Runner	Traffic Jumble
Платформи	Android	Android,	Android	Android
Мультиплеєр	+	-	-	-
3D графіка	+	+	-	+
Генеруємий рівень	+	+	+	+
Вибір скинів	+	+	-	+
Вибір карт	-	+	-	+
Можливість гальмувати	-	-	-	+

5

КОНЦЕПТ ГРИ

1. Головний герой: ігровий персонаж - автомобіль, що їде по зустрічній смузі.
2. Ігровий рівень: безкінечно генеруєма дорога з трьома смугами та перешкодами.
3. Перешкоди: ворожі машини їздять в зустрічному напрямку по всіх смугах та дорожны знаки.
4. Механіка: гравець повинен керувати автомобілем та уникати зіткнень з ворожими машинами, зібравши якомога більше бонусів на своєму шляху.
5. Управління: свайпами по екрану, та затисненням для гальмування на декілька секунд.
6. Бонуси: ігрові об’єкти що збираються головним героєм.
7. Рівні складності: кілька рівнів складності, які відрізняються швидкістю руху машин та кількістю перешкод.
8. Графіка: 3D графіка.

6

ВИМОГИ ДО ІГРОВОЇ СИСТЕМИ

1. Управління свайпами / утримуванням екрану декілька секунд.
2. Генерація нескінечного рівння, а саме дороги з розмішенням на ній ворожих машинок на шляху та перешкод.
3. Можливість збереження стану гри.
4. Можливість паузи гри.
5. Керування машинкою з можливістю гальмування та уникнення перешкод з завершення гри у разі зіткнення з перешкодою.
6. Наявність налаштуваннь з регулюванням гучності звуків та музики.
7. Наявність магазину де можна обміняти ігрові бонуси на нові карти або автомобілі.
8. Наявність рейтингу гравця.

7

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

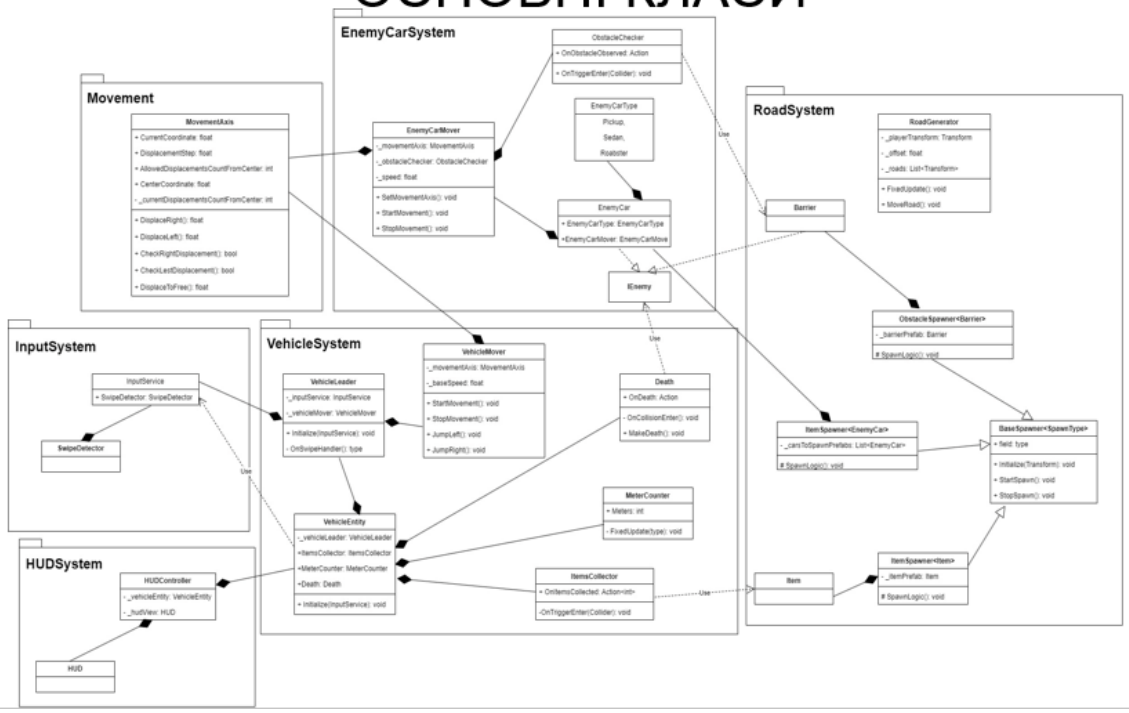
Rider

8

ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

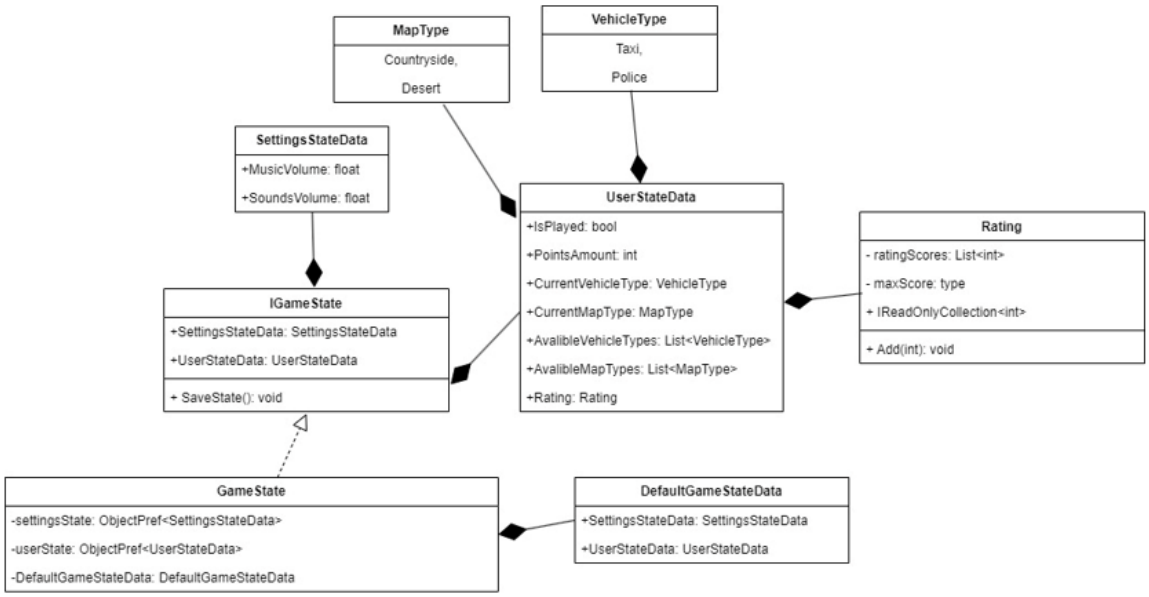
9

ОСНОВНІ КЛАСИ



10

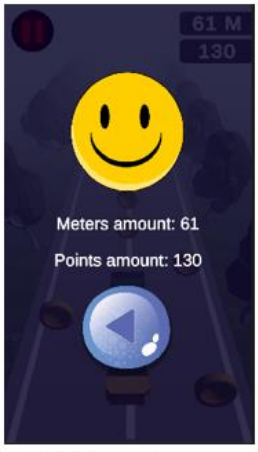
ДІАГРАМА КЛАСІВ ІГРОВОГО СТОРЕДЖУ



ЕКРАННІ ФОРМИ

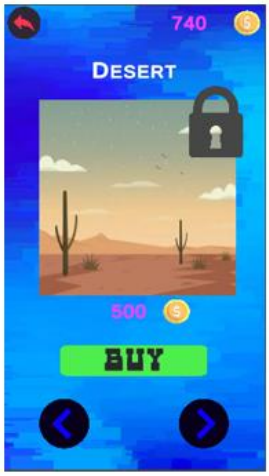
Головне меню

Рівень

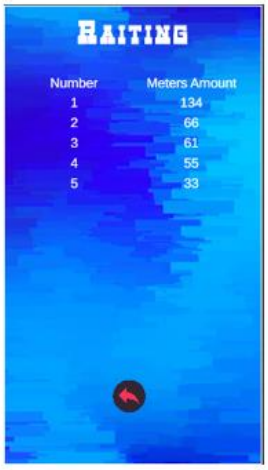


Вікно програшу

ЕКРАННІ ФОРМИ



Магазин рівнів



Рейтинг гравця

Налаштування

13

ВІДЕО ДЕМОНСТРАЦІЯ

14

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Осьмак В.Р. Твінери і Dotween – це що / Матеріали всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інфокомунікаційних технологіях». 20.04., ДУТ, м.Київ 2023 с. 125-126.

2. Осьмак В.Р. Сучасні інформаційні технології в україні і світі / IV Всеукраїнська науково-технічна конференція «Сучасний стан та перспективи розвитку IOT» 05.04., ДУТ, м.Київ 2023. 34-35.

15

ВИСНОВКИ

1. Проведено аналіз відеоігор і їх жанрів. Визначено потенційні ідеї для створення гри та обрано жанр Hyper Casual.
2. Проведено аналіз ринку гіперказуальких ігор для мобільних платформ та визначено типові механіки цього жанру.
3. Вивчено можливості ігрового рушія Unity. Визначено які можливості є в наявності для розробки гри та як їх можна використати в процесі розробки.
4. Спроектовано геймплей розроблюваної гри ключовою особливістю якого є механіка безкінечного руху з униканням перешкод та додано можливість гальмувати на деякий час.
5. Знайдено графіку та визначено дизайн гри.
6. Спроектовано архітектуру яка забезпечує гнучкість та можливості подальшого розширення проекту.
7. Розроблено ігровий додаток функціонуючим геймплеєм та іншими компонентами згідно концепту гри.
8. Проведено мануальне тестування гри, виявлено та виправлено деякі проблеми в роботі гри та забезпечено її більш стабільну роботу.

16

ДЯКУЮ ЗА УВАГУ!