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 МУЛЬТИПЛЕЄРНОЇ ГРИ FUNSHOT MULTIPLAYER В ЖАНРІ ШУТЕР З ВИКОРИСТАННЯМ ІГРОВОГО ДВИГУНА UNITY ДЛЯ ПЛАТФОРМИ ANDROID»

Виконав: студент 4 курсу, групи ПД-42

спеціальності

121 Інженерія програмного забезпечення

_____Осадчий Б.О.__

Керівник __Негоденко О.В___

Рецензент _____

Київ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти - «Бакалавр»

Спеціальність -121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Інженерії програмного забезпечення

_____Негоденко О.В.

«____»_____2021 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

ОСАДЧИЙ БОГДАН ОЛЕКСАНДРОВИЧ

1.Тема роботи: «Розробка мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android»

Керівник роботи: Негоденко Олена Василівна, доцент кафедри ІПЗ

затверджені наказом вищого навчального закладу від — «12» березня 2021 року № 65.

2. Строк подання студентом роботи 1.06.2021

3. Вхідні дані до роботи:

Алгоритм дії ігрових застосунків

Розробка мобільних застосунків для ОС Android

Тестування програмного забезпечення

Науково-технічна література, пов’язана з розробкою застосунків ОС Android

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

4.1 Аналітичні аспекти розробки мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android
4.2 Розробка мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android
4.3 Розробка архітектури мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android
4.4 Етапи реалізації мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android
4 5. Перелік графічного матеріалу
1._ Діаграма використання_____
2._ Діаграма прецедентів_____
3._ Простий ігровий цикл_____
4._ Компоненти об'єкта «Персонаж»___
6. Дата видачі завдання ___19.04.2021_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної леітератури	**20.09-29.09**	Виконано
2	Проблематика длослідження	**30.09-20.10**	Виконано
3	Аналіз існуючих 3 D ігор	**21.10-1.11**	Виконано
4	Дослідження програмних засобів	**2.11-20.11**	Виконано
5	Моделювання об’єкту проектування	**21.11-20.12**	Виконано
6	Розробка функціоналу застосунку	**21.12-1.29**	Виконано
8	Розробка презентації застосунку	**30.1-20.02**	Виконано
9	Попередній захист роботи	**11.05.2021**	
10	Подання роботи в деканат	**1.06.2021**	

Студент - _____

Керівник роботи _____ _____

РЕФЕРАТ

Текстова частина бакалаврської роботи 66 с., 36 рис., 2 табл., 43 джерела.

Ключеві слова: ANDROID STUDIO, UNITY, FUNSHOT, MULTIPLAYER, LAYOUT, XML

Об'єкт дослідження – процес проектування та розробки мультиплеєрної гри у жанрі Shooter.

Предмет дослідження – ігровий двигун UNITY та платформа Android.

Мета роботи – створення мультиплеєрної гри FunShot multiplayer в жанрі *Shooter* з використанням ігрового двигуна UNITY для платформи Android.

Методи дослідження – методи наукового пізнання такі як метод порівняння, аналогії, проведення експерименту, аналіз отриманих результатів.

Для досягнення поставленої мети в даній роботі:

1. Розроблено алгоритм для реалізації мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android;
2. Встановлено, що використання ігрового двигуна UNITY та середовища розробки Android Studio – це вдале рішення для досягнення поставленої мети.
3. Показано, що ігровий двигун UNITY зручний у використанні та надає можливість ефективної розробки застосунку на рівні з більш відомими двигунами.
4. На основі результатів виконаних досліджень розроблено мультиплеєрну гру FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

Практичне значення отриманих результатів полягає у отримані якісної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням

ігрового двигуна UNITY для платформи Android, яка може бути використана за потребою. Теоретичний матеріал може біти використаний під час викладання дисципліни за спеціальністю «Комп’ютерна інженерія».

В роботі виконано аналіз існуючих застосунків для операційної системи Android. Встановлено переваги та недоліки існуючих застосунків. В результаті аналізу було визначено основні потреби користувачів.

Проаналізовано можливості середовища розробки Android Studio. Розроблено логіку практичних завдань та загальну концепцію представлення інформації для користувачів застосунку.

Галузь використання – завдяки вільному доступу, застосунок може використовувати будь-яка людина, котра бажає грати у гру FunShot multiplayer в жанрі шутер.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, ТЕРМІНІВ, ОДИНИЦІ, ПОЗНАЧЕНЬ

API (від Application Programming Interface) – набір методів, констант і структур, які надаються ОС, додатком або фреймворком для використання в зовнішніх програмних продуктах і використовуваних програмістами при написанні своїх додатків.

GUI (Graphical User Interface) – графічний інтерфейс користувача – забезпечує можливість управління поведінкою системи через візуальні елементи управління – вікна, списки, кнопки, гіперпосилання.

TrueTypeFont (TTF) – формат зберігання шрифтів, розроблений компанією Apple в кінці 80-х років.

Engine – це набір систем, інструментів, які спрощують роботу програмісту при написанні гри

Shader – це програма для графічного конвеєра, яка використовується для завдання остаточних параметрів об'єктів і зображення.

Ассет – ресурс, що складається з однотипних даних, що є частиною ігрового контенту і використовуються, оброблюваний комп'ютерною грою.

ВСТУП

Актаульність дослідження. Комп'ютерна гра складається з різного роду компонентів, які розробляються багатьма програмістами, дизайнерами, сценаристами і іншими людьми, кожен з яких відіграє свою роль у проекті. Багатьох, комп'ютерні ігри просто рятують, тому що допомагають пережити тяжкі часи або стрес, коли поруч нікого з близьких людей немає. Однак з дітьми все складніше. Чи дійсно відеоігри їх розвивають, а якщо це так, то необхідно знати, скільки часу їм можна приділяти без збитку для здоров'я. США (університет штату Іллінойс) такі ж досліди провели з людьми старше 60 років. І виявилося, що для них відеоігри корисні – вони сприяють поліпшенню робочої (короткочасної) пам'яті, прискорюють мозкову діяльність і т.д. Але проблема залишається в дітях. Цим питанням навіть займався Європейський парламент. Європарламент не знайшов доказів того, що відеоігри шкідливі для дітей. На їхню думку, навіть жорстокі ігри корисні для дітей. Багато ігор розвивають стратегічне, творче, нестандартне мислення, а групові ігри вчать взаємодії між людьми. Однак психологи не прийшли до єдиної думки про вплив відеоігор на дітей. Деякі з них вважають, що такі ігри провокують агресію. На думку інших, багато ігор допомагають розвивати як розумові здібності, так і соціальні навички.

Мета та завдання. Метою дипломної роботи є створення мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

Для досягнення мети дипломної роботи потрібне рішення таких завдань:

- розкриття теоретичних аспектів створення якісної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android;
- формування проблематики дослідження та постановка завдань;
- розробка основних модулів мультиплеєрної гри FunShot

multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android та опис структурних компонентів;

- тестування створеної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

Об'єктом роботи виступає процес проектування та розробки мультиплеєрної гри у жанрі Shooter.

Предмет – ігровий двигун UNITY та платформа Android.

Методи дослідження. Під час написання дипломної роботи використовувалися такі методи наукового пізнання як; метод порівняння, аналогії, проведення експерименту, аналіз отриманих результатів.

Практичне значення отриманих результатів. Практичне значення отриманих результатів полягає у отримані якісної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android, яка може бути використана за потребою. Теоретичний матеріал може біти використаний під час викладання дисципліни за спеціальністю «Інженерія програмного забезпечення».

Структура роботи. Структуру роботи складають: перелік скорочень, вступ, чотири розділи, висновки, перелік використаної літератури та додатки. Загальний обсяг роботи 66 сторінок.

РОЗДІЛ 1
ТЕОРЕТИЧНІ АСПЕКТИ РОЗРОБКИ МУЛТИПЛЕЄРНОЇ ГРИ FUNSHOT MULTIPLAYER

1.1 Поняття та генезис 3D ігор

Комп'ютерна гра (3D, англ. Three-dimensional) – гра, візуальний простір якої цілком побудовано з тривимірних об'єктів. Персонаж знаходиться в тривимірному просторі і в деяких іграх має повну свободу пересування.

Тривимірні ігри принципово відрізняються від двомірних (2D), в яких ігровий простір складається з готових плоских зображень – мальованих від руки або створених художником в тривимірному середовищі моделювання і потім збережених як звичайні зображення. Завдання комп'ютера зводиться тут лише до комбінування цих зображень відповідно до алгоритмів гри, чим пояснюються низькі вимоги двомірних ігор до потужності комп'ютера. Свобода пересування персонажа в двомірних іграх обмежена: він може пересуватися лише за передбаченим розробниками маршруту.

Можливі проміжні варіанти між 2D і 3D. Іноді для них використовується термін 2.5D.

Простір гри трьохмірний, але моделі ігрових об'єктів – будівель, рослин, персонажів – представляють собою плоскі картинки, спрайт. Приклад – ранні тривимірні ігри, такі як Wolfenstein 3D і Doom. В сучасних тривимірних іграх спрайт використовуються лише для відображення об'єктів, присутніх в дуже великій кількості (наприклад, трави та дерев), а також об'єктів, віддалених від персонажа на велику відстань. Робиться це для економії обчислювальних ресурсів комп'ютера: наприклад, моделювання повністю тривимірного дерева з тисячами листів – дуже складне завдання для комп'ютера, а моделювання цілого лісу – завдання нездійсненне, і без спрайтів тут обійтися неможливо.

Простір гри двохмірний, але моделі персонажів виконані тривимірними. Така технологія використовується в багатьох квестах. Вона дозволяє поєднати високоякісний двомірний мальований фон (як правило, частково анімований) з «живими» персонажами, що вільно рухаються і при цьому невимоглива до потужності комп'ютера. Яскравий приклад такої гри – Syberia.

Управління в тривимірних іграх, як правило, здійснюється одночасно клавіатурою і мишкою. За допомогою кнопок змушує персонажа рухатися в потрібному напрямку (зазвичай це клавіші W, S, A, D, рідше – клавіші-стрілки). Рух миші змушує персонажа повертатися. Клацання лівою кнопкою миші відповідає якому-небудь дії – в залежності від жанру гри це може бути стрілянина, удар мечем, взаємодія з предметом або іншим персонажем. Клацання правою кнопкою миші, як правило, в шутерах і рольових іграх відповідає блокуванню атаки або альтернативній атаці, а в квестах – відкриття меню. Клавіша Space зазвичай відповідає стрибку, а одночасне натискання клавіш Shift і W змушує персонажа бігти. Призначення інших клавіш клавіатури індивідуально для кожної гри.

1.2 Класифікація 3D ігор

Графіка на мобільних пристроях по стилю не відрізняється від графіки на ПК або консолях. Просто на більш потужних платформах більш присутній стиль реалізму, а в мобільних іграх мультяшний стиль.

У реалізмі виходять майже всі тріпл-А проекти. Причому сеттинг може бути яким завгодно, будь то sci-fi, вестерн або наш час. Містобудівники, шутери, симулятори і т.д.

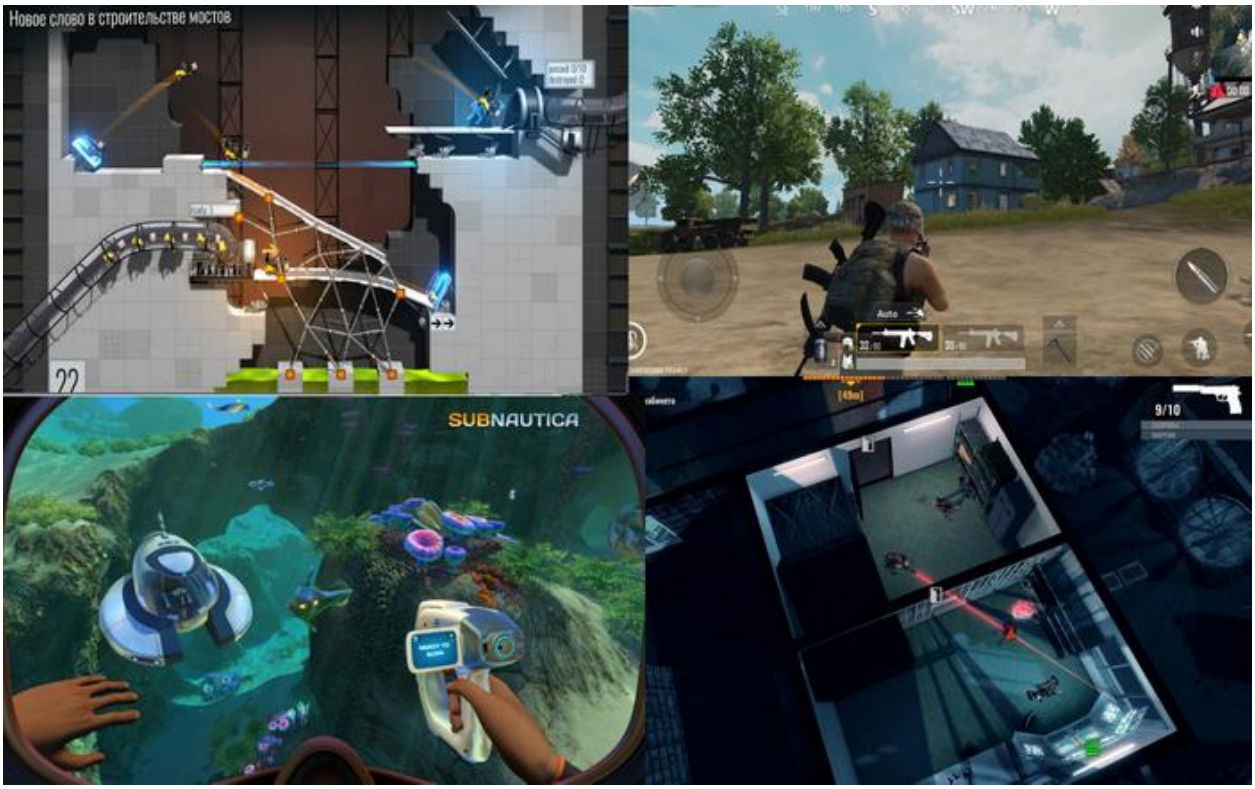


Рисунок 1.1 Реалізм у 3D грі

Реалізація може бути різна. Portal bridge constructor спірний момент там чоловічки не дуже то реальні, але інші матеріали досить реалістичні.

"Low Poly Art"

Це ціла релігія. Люди люблять low poly. У 2017 році був бум лоу полі. Але і зараз виходять проекти в цьому стилі, досить високої якості. Правда часто його змішують з іншими стилями або якось видозмінюють.

Лоу полі характеризується найчастіше низькополігональними моделями без згладжування. Тобто чітко видно кожен полігон. Текстури не використовують. Кожен полігон має один колір. Так і фарбують. Розробники імпровізують і часто виходять прекрасні візуальні поєднання. Складність моделей варіюється. Але переважає мінімалізм, "чистота" моделей.

Рисунок 1.2 Low Poly Art

"Hand Painted"

Ще одна релігія. Найчастіше цей стиль використовують у фентезі. Доту, варкрафт і т.д. Але не тільки цими іграми характеризується даний стиль. Він може бути з ухилом в комікси, мультики або навіть реалізм.

Характеризується він малюванням текстур від руки. Всі тіні, відблиски, а іноді і дрібні деталі малюються на текстурі без геометрії. Хенд Пейнтед може бути різний).

"Cartoon" або мультяшний (казуальний)

По суті і лоу поли і хенд пейнтед може бути віднесений до цієї групи. Буває текстури для такої графіки малюють від руки, буває не використовують текстури зовсім.

Рисунок 1.3 Hand Painted

Рисунок 1.4 Cartoon

Відмінними рисами буде барвистість, відносна простота, перебільшення акцентів і нестандартні пропорції. Така, грань дуже тонка, але все таки такі ігри відрізняються і від лоу поли і від хенд пейнтед. У кожного автора свій власний стиль. І гри в такий стилізації набувають неповторний стиль автора.

Як мультики: Рік і Морті по стилю відрізняються від Міккі Мауса або звероподіса проте вони є мультиками.

Але все ж в казуальній графіці можна виділити окремі підстилі з яскраво-вираженими відмінними рисами.

Рисунок 1.5 Воксельний стиль

Воксельний стиль – якийсь піксель арт в 3D. Все складається з вокселей (тривимірний куб одного кольору). Теж є варіації, коли вокселі мають текстуру, як в майнкрафт.

Рисунок 1.6 Пластиліновий стиль

Рисунок 1.7 3D pixel art

3D pixel art. Як правило нізкополігональні моделі з піксельною текстурою.

Рисунок 1.8 Little big planet

Стиль little big planet – це картонні і тканинні фігури.

Оригінальний візуальний стиль завжди знаходиться у виграшному положенні в порівнянні з тисячами однакових ігор. Оригінальність може виражатися в незвичайному малюванні текстур, в незвичайних рисах персонажів, в незвичайній геометрії моделей і так далі.

1.3 Проблематика та постановка завдань

Проаналізувавши проблему впливу відеоігор на розвиток дітей варто відзначити, що відеоігри можуть розважати і розвивати дитину. Однак вони віднімають час, який вони могли б витратити на інші заняття. За даними досліджень, проведених в Україні, підлітки, що захоплюються відеоіграми, витрачають на домашні завдання приблизно на 40% менше часу, ніж це було б потрібно, а на читання – на 30% менше часу. Тому справа батьків – регулювати час, який діти проводять за відеоіграми. Таким чином, відеоігри можуть бути корисними, а можуть і нашкодити. Якщо діти займаються ними в розумних межах, це сприяє, поряд з іншою діяльністю, розумовому

розвитку і допомагає засвоювати нові навички. Перш за все, дитина набуває і вдосконалює навички роботи з комп'ютером, що стане в нагоді йому при подальшому навчанні. Правильно підібрані відеоігри розвивають у дітей мислення, увагу і логіку, пам'ять і здатність приймати рішення. Управління відеогрою покращує координацію рухів дитини. Навчальні ігри допомагають підготувати його до школи. Групові відеоігри вчать терпінню і взаємодії в групі. Деякі ігри розвивають творчу уяву.

Отже, ми розуміємо, що потрібно зробити застосунок, котрий допоможе у розвитку дітей, координації рухів та підвищенню навичок роботи за комп’ютером. Для досягнення цілі:

- розробимо основні модулі мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android та опис структурних компонентів;
- проведемо тестування створеної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

РОЗДІЛ 2
РОЗРОБКА МУЛТИПЛЕЄРНОЇ ГРИ FUNSHOT MULTIPLAYER

2.1 Аналіз задачі

Технології розвиваються з величезною швидкістю. Кожен день з'являються на світ все нові і нові методи рішень, мови програмування, системи та багато іншого. І не виняток з цього – це комп'ютерні ігри. Більшість людей, що стикалися з комп'ютерними іграми, мають про це проведення часу або негативну, або позитивну думку. Аналітична компанія Newzoo, яка займається дослідженням ринку відеоігор в різних країнах, опублікувала результати своїх досліджень, проведених на території України.

У нашій країні, судячи з усього, на сьогоднішній день проживає понад 25 мільйонів людей, що грають в ігри. В даному випадку вважаються ті люди, які регулярно грають у відеоігри будь-якого жанру і на будь-яких платформах. Важко назвати граючою ту людину, яка цим питанням зовсім не цікавиться і доторкається до ігор один-два рази на рік у свята. 98 відсотків від загальної кількості гравців нашої країни, так чи інакше, грають в ігри на своєму персональному комп'ютері. Кожна людина сама для себе вирішує, що для нього важливо в грі, будь то сюжет, ігровий процес, графіка, співтовариство. Так само дуже важливим фактором є атмосфера, яка виникає при грі в ту чи іншу гру.

Закінчення школи, вступ до університету, перемоги на змаганнях, підвищення максимальної кількості підтягувань – все це для кожної людини досягнення.

Для того, щоб у гру було цікаво грати, необхідний ретельно пророблений прогрес розвитку персонажа. У різних іграх, цей прогрес реалізований по-різному, але в основному він присутній у всіх іграх, тільки в різній кількості інформативності. Так, наприклад, в іграх жанру RPG система прогресу розвитку персонажа є невід'ємною частиною самої гри. Кількість

досвіду, необхідного для підвищення рівня персонажа, кількість здоров'я / мани / ресурс, досягнення про різні види дій в ігровому світі, система завдань, інформація про спорядження і багато іншого. Перераховувати можна дуже довго, саме тому ця частина гри є дуже важливою.

2.1.1 Опис гри та правила

Ігровий мир являє собою двомірне поле (на кожному рівню різне) по якому переміщується ігрок.

Гра є мультиплеєрною в жанрі шутер. На кожному рівні є перешкоди та противники в яких необхідно стріляти.

2.2.2 Вимоги та не функціональні характеристики

Вимоги:

- Мультиплеєрна гра – основна ідея полягає у знищенні противників шляхом обстрілу, багатокористувальний режим передбачено.

- Гра повинна бути інтуїтивно зрозумілою основній масі не досвідчених гравців.

- Ігровий процес не повинен бути занадто складним, ніяких зайвих деталей відволікаючих гравця.

- Графічна частина повинна бути не сильно перевантажена, текстури не повинні «різати» очі, повинна бути використана спокійна колірна гамма.

- Контент повинен легко сприйматися і бути зрозумілим гравцеві в будь-який момент часу гри. Ігрове меню і ігрове поле витримані в єдиному стилі і не повинні сильно відрізнятися в управлінні.

- Ігровий додаток повинен мати можливість запускатися на будь-якому Windows сумісному IBM PC сумісному комп'ютері.

Не функціональні характеристики:

Плавність пересування об'єктів, відсутність ривків є хорошим показником продуктивності движка.

Ігровий додаток повинен мати гнучку архітектуру. Програміст повинен мати можливість легко розширювати можливості ігрового додатку, допрацьовувати логіку гри (змінювати властивості існуючих в грі об'єктів, додавати нові об'єкти).

У користувача не повинно бути проблем із запуском програми на різних підтримуваних платформах.

2.2 Вибір ігрової платформи

Ігровий движок являє собою набір підсистем, контролюючих певні частини гри. Більшість з ігрових движків мають наступні підсистеми:

- графічна підсистема;
- підсистема вводу-виводу;
- звукова підсистема;
- мережева підсистема;
- ядро.

Використання вбудованих в движок візуального редактора, готових модулів рендеринга, анімації спрайтів і обробки зіткнень спрощує процес розробки. Багатоплатформовий движок зробить можливим перенесення вашого застосування на іншу апаратну платформу, без будь-яких серйозних змін в коді. Передбачене в складі движка інтегроване середовище розробки дозволить зробити код більш читабельним, зрозумілим і організованим.

Плюси використання кроссплатформенних мобільних додатків:

- Можливість використовувати один і той же код в інших проектах і на інших операційних системах знову і знову.

- Доступ розробників до плагінів і модулів, можливість вбудовування в інші сервіси, інструменти.

- Компаніям тепер немає сенсу наймати розробників під різні платформи. Якщо вони будуть розробляти кроссплатформний додаток, то

вони можуть, використовуючи одну команду програмістів, охопити кілька платформ.

- Можливість інтеграції в хмарні сервіси.

- Простота розгортання прискорює процес розробки мобільного додатку.

- Додатки вимагають, мобільної реалізації охоплюють такі категорії як Фото, Відео, Новини, Бізнес, Ігри, Книги та Розваги

Таким чином найбільш підходящим для розробки будь-якого сучасного додатку, є кросплатформені рішення, які забезпечують високу швидкість роботи на широкому колі популярних мобільних платформ і зручність розробки.

Огляд існуючих на даний момент ігрових движків:

- Unreal Engine, один з найбільш просунутих ігрових движків. Може бути використаний абсолютно безкоштовно, до того моменту коли дохід перевищить 3000 $.

- CryEngine найпопулярніший комерційний движок, на якому були зроблені такі ігри як FarCry, Sniper2, Giant, Ryse: Son of Rome і інші.

- Phaser HTML5 движок, який має відкритий вихідний код. хороший варіант для тих, хто створює web гри.

- Source 2. Движок був анансований ще на GDC 2015. Є приймачем движка Source від Valve. Був використаний в таких відомих іграх як Counter-Strike, Half-Life 2 і безліч інших.

- Construct 2 дозволяє створювати ігри, не написавши при цьому, жодного рядку коду. Для використання повного функціоналу потрібно придбати ліцензію.

- Corona SDK надає інструментарій для створення кроссплатформенних ігрових додатків. Є можливості для монетизації за допомогою Corona Ads.

- Godot Engine, ще один безкоштовний ігровий движок, добре зарекомендував себе в середовищі інді-розробників. Має потужний

візуальний редактор сцени, редактор анімації, своє інтегрованє середовище розробки.

- Unity один з найкращих на сьогоднішній день ігрових движків. Безкоштовна версія може використовуватися для створення як 2D таки 3D ігор. Охоплює 24 платформи: це і десктопи, і консолі, веб платформи, мобільні пристрої.

Для невеликих інді-компаній або окремих розробників принципово щоб інструментарій для створення ігор коштував дешево, щоб не мав зайвого функціоналу, був простий в освоєнні. Для них є важливими такі аспекти як:

- Кросплатформенність. Для забезпечення доступності додатку на як можна більшій кількості платформ, і переносимість коду без істотних змін.

- Движок повинен бути розширюємим, бути модульним.

Можливість освоїти движок в максимально короткі терміни.

Розробка ігрового програми не повинна вимагати істотних фінансових вливань.

- Не висока кваліфікація розробників не повинна бути проблемою для створення програми.

Таким чином, провівши дослідження сучасних ігрових движків прийнято рішення використовувати Unity3D.

2.3 Опис та характеристики ігрової платформи

Ігрова платформа Unity3D складається з декількох вікон, кожне з яких виконує ту чи іншу задачу. В основному це вікна:

1) Ієрархія (Hierarchy);

2) Сцена (Scene);

3) Ігрове вікно (Game);

4) Оглядач (Inspector);

5) Проект (Project);

6) Панель інструментів (Toolbar);

7) Консоль (Console).

1. Ієрархія (Hierarchy)

Ієрархія (Hierarchy) містить всі об'єкти (GameObject) в поточній сцені. Деякі з них є прямими екземплярами файлів Ассет, таких як 3D-моделі, а інші – екземпляри префабів, призначених для користувача об'єктів, з яких складається велика частина нашої гри. Так само Unity використовує концепцію під назвою Parenting. Тобто якщо ми перетягнемо, деякий об'єкт на інший, ми створимо батьківський зв'язок. Щоб побачити дочірні елементи батьківського об'єкта, досить використовувати розкриваючу стрілку навпроти родового об'єкта. При додаванні і видаленні об'єктів в сцені, вони також будуть з'являтися і зникати з ієрархії.

Ігрове вікно (Game)

Ігрове вікно – це вид в грі. Це відображення готової, фінальної гри. У цьому режимі можна подивитися на те, як це виглядало, якщо гра вже була б закінчена.

Для того щоб увійти в цей режим, необхідно на панелі інструментів натиснути клавішу Play. На рис. 2.1 можна побачити елементи переходу в ігровий режим. Всі зміни, внесені в гру поки Play активний, будуть лише тимчасовими і скинуться, коли відбудеться вихід з цього режиму.

Рисунок 2.1 Елементи переходу в ігровий режим

Так само при активному режимі Play, ми можемо переходити в режим Scene, щоб подивитися більш детально, що конкретно відбувається в даний момент.

Панель інструментів (Toolbar)

Рисунок 2.2 – Вікно інструментів

На рис. 2.2 показано вікно інструментів. Вибрати інструмент можна натиснувши ліву кнопку миші, або відповідні клавіші на клавіатурі: W для переміщення, E для обертання і R для маштабування.

Unity може працювати з 3D моделями будь-якої форми, що створюються в додатках для моделювання. Однак існує ряд примітивних моделей, які можна створити прямо в Unity: Куб (Cube), Сфера (Sphere), Капсула (Capsule), Циліндр (Cylinder), Площина (Plane) і Квад (Quad). Ці об'єкти часто застосовуються, як є (площину зазвичай використовується в якості поверхні землі, наприклад), але також вони дозволяють швидко створити прототипи складних об'єктів з метою тестування. Будь-який з примітивів може бути доданий в сцену за допомогою відповідного пункту в меню GameObject 3D Object. Крім всіх примітивів, в Unity існує ландшафтний движок. Завдяки йому у користувача є величезний вибір різних інструментів для подальшого редагування ландшафту. Завдяки цьому движку, можна зробити різні висоти, додати рослинність, накласти текстури та інше. Щоб додати цей елемент необхідно вибрати пункт в меню GameObject 3D Object Terrain.

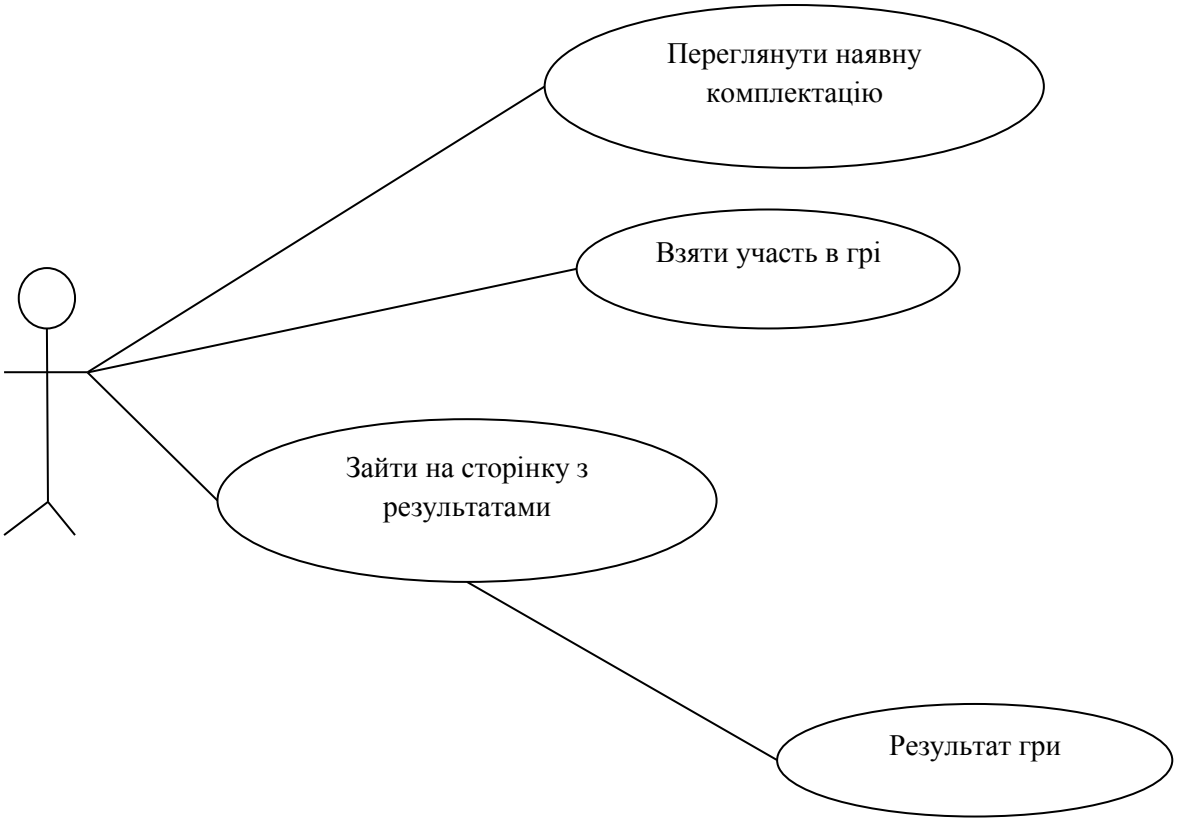


Рисунок 2.3 – Діаграма використання

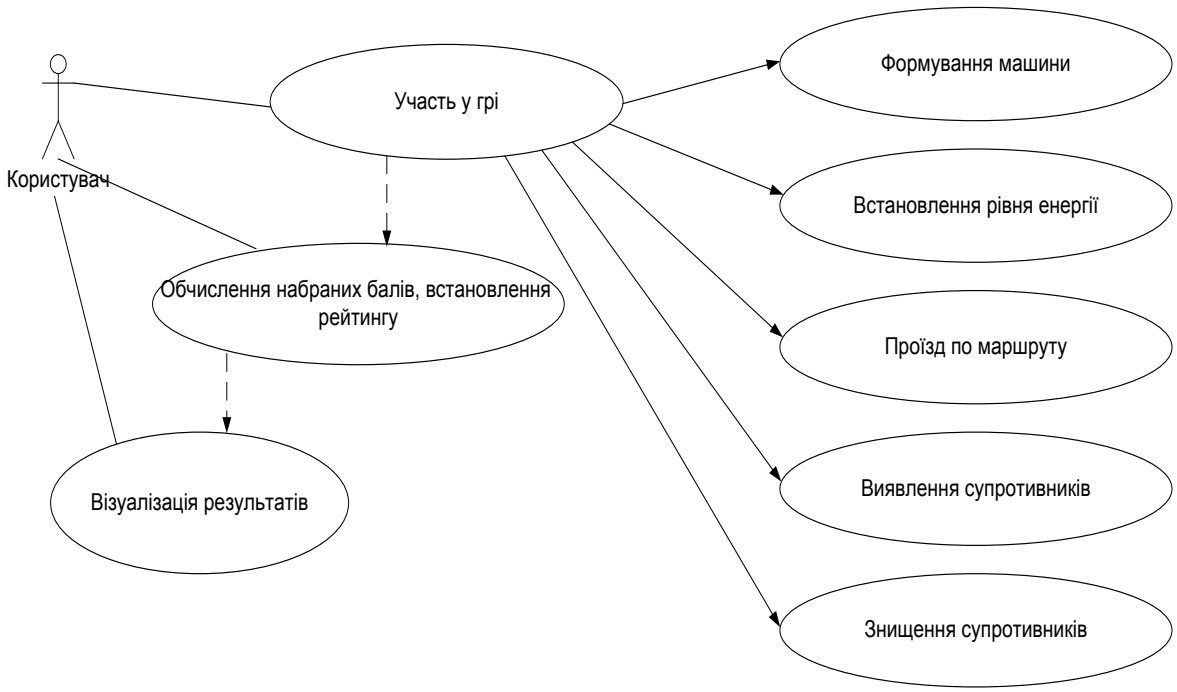


Рисунок 2.4 – Діаграма прецедентів

РОЗДІЛ 3
АРХІТЕКТУРА

3.1 Вибір програмного забезпечення

Для початку необхідно визначитися, на яких пристроях буде функціонувати система. Для цього доцільно висунути ряд вимог до апаратної платформи і знайти рішення, яке найбільшою мірою їм задовольняє.

По-перше, пристрій повинен надавати оперативний і надійний доступ до додатку.

По-друге, необхідна наявність досить потужного процесора і оперативної пам'яті для здійснення необхідних обчислень і досягнення необхідної швидкодії. Вбудована пам'ять, використовувана для зберігання операційної системи, програмного комплексу і користувальницьких даних повинна бути досить великою. Бажано наявність слоту для підключення карти пам'яті з метою збільшення обсягу пам'яті.

По-третє, пристрій повинен так само володіти хорошим дисплеєм для відображення інформації в денний і нічний час. Батарея повинна тримати заряд досить тривалий час.

Всім перерахованим вимогам достатньою мірою задовольняють цифрові пристрої, звані комунікаторами або смартфонами.

Про вибір операційної системи мова піде далі.

Операційна система Symbian, яка беззастережно лідирувала на ринку до 2010 року в даний час майже повністю втратила свої позиції. Спершу Sony Ericsson оголосив про припинення використання Symbian, щоб зосередитися на випуску пристроїв під управлінням Android. Слідом за ним Samsung припиняє підтримку Symbian і переходить на використання Android, Windows Phone 7 і Bada, ОС власної розробки.

Навесні 2011 року сама Nokia оголосила про те, що Windows Phone 7 стане ключовою платформою для смартфонів, проте компанія не планує відмовлятися від платформи Symbian і стане підтримувати мобільну платформу Symbian «щонайменше» до 2020 року.

Незважаючи на це, перспектив у якісному розвитку даної операційної системи не передбачається. І, незважаючи на те, що розробка автоматизованої системи можлива і в рамках даної платформи, від цього варіанту варто відмовитися.

Apple iOS (до 2010 року відома як iPhone OS) мобільна операційна система, розроблена американською компанією Apple на основі Mac OS X спочатку для iPhone, а потім розширена для підтримки таких мобільних пристроїв, як Apple iPod, iPad і Apple TV і в даний час динамічно розвивається.

Розробка додатків для iOS ведеться в середовищі XCode, який підтримує мови C, C + +, Objective-C, Objective-C + +, Java, AppleScript, Python і Ruby з різними моделями програмування.

Apple не ліцензує iOS для установки на стороннє обладнання, а ціна комунікатора iPhone в Україні досить велика. Цей факт безумовно сильно ускладнює впровадження автоматизованої системи, розробленої для iOS з фінансової точки зору. Тому дана ОС так само не була обрана.

Набирає деяку популярність операційна система Windows Phone 7, проте вона має низку недоліків, серед яких необхідно виділити відсутність вбудованої підтримки баз даних. SQL Server Compact не включено до складу WP7. Також не включені SQLite або будь-які інші сторонні СУБД. Відсутня так само підтримка взаємодії між процесами (IPC) і немає підтримки сокетів.

Продовжуючи аналізувати ринок операційних систем для смартфонів, можна прийти до доцільності використання ОС Android.

Операційна система Android – портативна (мережева) операційна система для комунікаторів, планшетних комп'ютерів, цифрових програвачів, наручних годинників. Спочатку розроблялася компанією Android Inc., яку

потім купила Google. Згодом Google ініціювала створення альянсу Open Handset Alliance (OHA), який зараз і займається підтримкою і подальшим розвитком платформи.

Частки ринку, які займали операційні системи в певні квартали, наведені в таблиці 3.1.

Таблиця 3.1 – Частки ринку операційних систем для смартфонів

Квартал	Android	iOS	Symbian	Microsoft	Bada	RIM	Other
2016 Q2	0%	5,20%	62,30%	11,90%	0%	12,90%	20,60%
2016 Q3	0%	4,60%	49,50%	10,40%	0%	13,10%	35,50%
2016 Q4	0%	2,80%	57,50%	12,10%	0%	15,40%	27,60%
2017 Q1	0,60%	13,10%	50,30%	11,20%	0%	18,20%	24,80%
2017 Q2	1,10%	10,60%	46,50%	12,20%	0%	19,70%	29,60%
2017 Q3	1,60%	10,50%	48,80%	10,20%	0%	19,70%	28,90%
2017 Q4	1,80%	13,00%	51,00%	9,30%	0%	20,50%	24,90%
2018 Q1	3,40%	17,00%	44,20%	7,90%	0%	19,00%	27,50%
2018 Q2	7,60%	16,20%	44,70%	7,90%	0%	20,60%	23,60%
2018 Q3	9,60%	15,30%	44,20%	6,80%	0%	19,30%	24,10%
2018 Q4	17,20%	14,20%	41,20%	5,00%	0,90%	16,10%	21,50%
2019 Q1	25,30%	16,60%	36,30%	2,80%	1,10%	17,50%	17,90%
2019 Q2	31,10%	16,10%	32,90%	3,40%	1,30%	11,60%	15,20%
2019 Q3	36,00%	16,80%	27,40%	3,60%	1,70%	10,90%	14,50%
2019 Q4	43,40%	18,20%	22,10%	1,60%	1,90%	9,70%	12,80%
2020 Q1	52,50%	15,00%	16,90%	1,50%	2,20%	8,90%	11,90%
2020 Q2	56,90%	18,90%	11,70%	1,90%	2,10%	8,70%	8,50%

Операційна система Android має цілу низку переваг [13]:

По-перше, це повна відкритість системи. Розробка додатків доступна всім бажаючим, і тому Android є однією з найбільш легко розширюваних платформ.

По-друге, система повністю універсальна. Android OS можна встановлювати і на смартфони, і на кишенькові комп'ютери, і навіть на телефони (у полегшеному вигляді).

Це одна з найбільш зручних і практичних мобільних операційних систем, що використовуються в сучасному світі і її настроювання дуже просте.

Популярність операційної системи Android стала наслідком того, що провідні компанії мобільних телефонів, наприклад, HTC, Motorola, Samsung Electronics, і багато інших виробників стали використовувати Android на своїх смартфонах.

Перевагою Android від Google також є так само і те, що нова версія Android запускається майже кожні 3 місяці. Оновлення телефону новими версіями допомагає користувачам насолоджуватися більш якісними послугами.

Перед тим, як приступити до розробки програми для Android, необхідно ознайомитися з архітектурою системи та основними особливостями цієї платформи.

Система Android – це програмний стек для мобільних пристроїв, який включає операційну систему, програмне забезпечення проміжного шару (middleware), а також основні користувальницькі додатки.

Архітектуру Android прийнято ділити на чотири рівні [1]:

- Рівень ядра;
- Рівень бібліотеки середовища виконання;
- Рівень каркаса додатків;
- Рівень додатків.

На рисунку 3.1 показані основні компоненти операційної системи та їх взаємодія між собою.

Кожна програма в ОС Android запускається у власному примірнику віртуальної машини Dalvik. Таким чином, всі працюючі процеси ізольовані від операційної системи і один від одного. І взагалі, архітектура Android Runtime така, що робота програм здійснюється строго в рамках оточення віртуальної машини. Завдяки цьому здійснюється захист ядра операційної системи від можливої шкоди з боку інших її складових. Тому код з

помилками не зможе зіпсувати Android і пристрій на його базі, коли вони спрацюють. Така захисна функція, разом з виконанням програмного коду, є однією з ключових для надбудови Android Runtime.

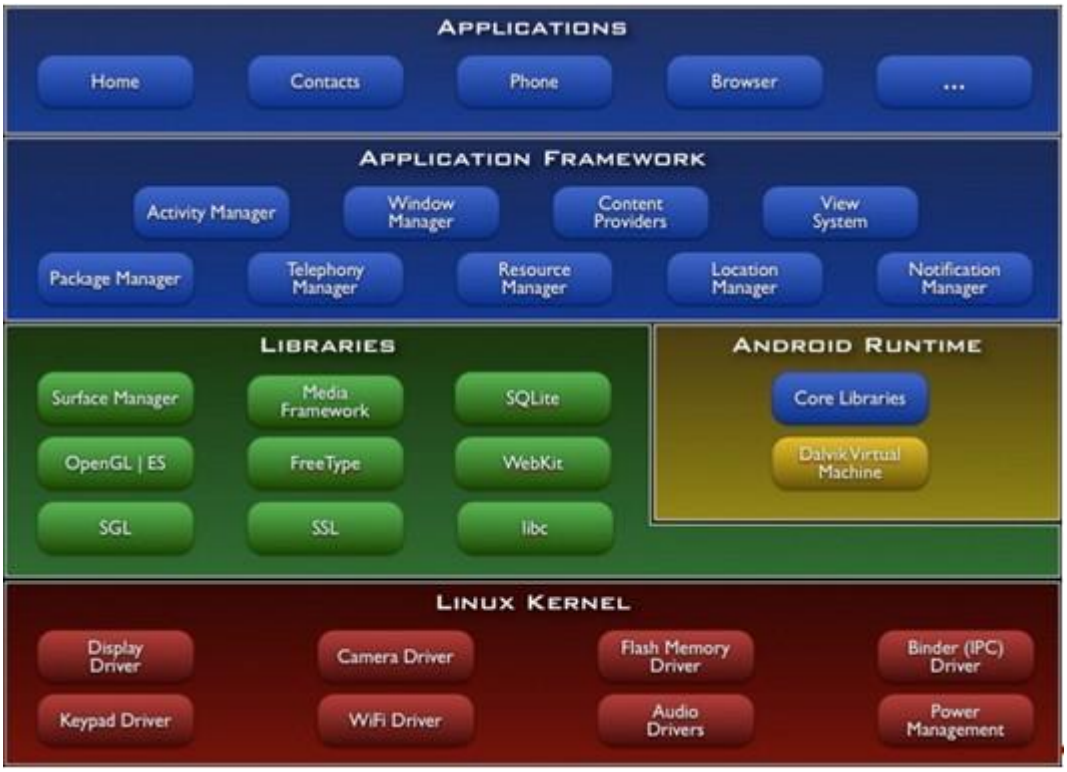


Рисунок 3.1 Архітектура операційної системи Android

Рівнем вище розташовується Application Framework, іноді званий рівнем каркаса додатків. Саме через каркаси додатків розробники отримують доступ до API, що надаються компонентами системи, що лежать нижче рівнем. Крім того, завдяки архітектурі фреймворку, будь-якого додатка надаються вже реалізовані можливості інших додатків, до яких дозволено одержувати доступ. У базовий набір сервісів і систем, що лежать в основі кожної програми і є частинами фреймворку, входять [2]:

- Багатий і розширюваний набір уявлень (Views), який може бути використаний для створення візуальних компонентів додатків, наприклад, списків, текстових полів, таблиць, кнопок або навіть вбудованого web-браузера.

- Контент-провайдери (Content Providers), керуючі даними, які одні програми відкривають для інших, щоб ті могли їх використовувати для своєї роботи.

- Менеджер Ресурсів (Resource Manager), що забезпечує доступ до ресурсів без функціональності (що не несуть коду), наприклад, до строковим даними, графіку, файлів та інших.

- Менеджер Оповіщень (Notification Manager), завдяки якому всі програми можуть відображати власні повідомлення для користувача в рядку стану.

- Менеджер Дій (Activity Manager), який управляє життєвими циклами додатків, зберігає дані про історію роботи з діями, а також надає систему навігації по них.

- Менеджер Місця розташування (Location Manager), що дозволяє додаткам періодично отримувати оновлені дані про поточний у географічному положенні пристрій.

Таким чином, завдяки Application Framework, додатки в ОС Android можуть отримувати в своє розпорядження допоміжний функціонал, завдяки чому реалізується принцип багаторазового використання компонентів додатків і операційної системи. Природно, в рамках політики безпеки.

На вершині програмного стека Android лежить рівень додатків (Applications). Сюди належить набір базових додатків, який встановлений на ОС Android. Наприклад, в нього входять браузер, поштовий клієнт, програма для відправки SMS, карти, календар, менеджер контактів та багато інших. Список інтегрованих програм може змінюватися в залежності від моделі пристрою та версії Android. І крім цього базового набору до рівня додатків відносяться в принципі всі програми під платформу Android, в тому числі і система, що розробляється.

Вибір пристроїв-комунікаторів в якості апаратної основи для реалізації проекту і Android OS в якості операційної системи значною мірою зумовив і

те, який інструментарій використовуватиметься для розробки і тестування додатків.

Система розроблялася на об'єктно-орієнтованій мові Java, розробленій компанією Sun Microsystems, яка в даний момент придбана корпорацією Oracle. Вибір даної мови був зроблений з низки міркувань.

Програми на мові Java транслюються в байт-код, що виконується віртуальною машиною (Java Virtual Mashine) - програмою, яка займається обробкою байт і передавальної інструкції обладнанню як інтерпретатор.

Гідність даного способу виконання програм полягає в повній незалежності байт-коду від операційної системи та обладнання, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина. На комунікаторах з операційною системою Android таку функцію виконує віртуальна машина Dalvik.

Повний контроль виконання програми віртуальною машиною виливається в організацію гнучкої системи безпеки. Будь-які операції, які перевищують встановлені повноваження програми, викликають колізії при роботі з даними) викликають негайне переривання, яке може бути відповідно оброблено.

До недоліків концепції віртуальної машини можна віднести те, що виконання байт-коду віртуальною машиною в деякій мірі знижує продуктивність програм і алгоритмів, реалізованих на мові Java. Але і вплив цих особливостей зведено до мінімуму, завдяки застосуванню технології трансляції байт-коду в машинний код безпосередньо під час роботи програми (JIT-технологія) і широкому використання переносного орієнтованого коду в стандартних бібліотеках.

Система Java створювалася об'єктно-орієнтованою з самого початку. Об'єктно-орієнтована парадигма в чому найбільш зручна при створенні програмного забезпечення.

Потрібно відзначити, що існує можливість розробляти програми і на C / C + + (за допомогою Native Development Kit), і на Basic (за допомогою

Simple) і з використанням інших мов. Також можна створювати власні програми за допомогою конструкторів додатків, таких як App Inventor.

Для розробки програм на мові Java необхідно спеціальне програмне забезпечення.

Java Development Kit (JDK) - безкоштовно розповсюджуваний корпорацією Oracle Corporation комплект розробника додатків мовою Java, що включає в себе стандартні бібліотеки класів Java, приклади, документацію, різні утиліти і виконавчу систему Java (JRE).

Варто відзначити такий важливий факт, що до складу JDK не входить інтегроване середовище розробки, яка встановлюється окремо.

Всі сучасні інтегровані середовища розробки додатків на Java, такі, як NetBeans IDE, Sun Java Studio Creator, IntelliJ IDEA, Borland JBuilder, Eclipse, спираються на сервіси, що надаються JDK. Більшість з них для компіляції Java-програм використовують компілятор з комплекту JDK. Тому ці середовища розробки або включають в комплект поставки одну з версій JDK, або вимагають для своєї роботи попередньої інсталяції JDK на машині розробника.

Незважаючи на те, що для розробки можна використовувати цілий ряд сучасних інтегрованих засобів розробки (IDE), список яких наведено в попередньому параграфі, вибір припав на Eclipse. Це відбулося з наступних причин:

Eclipse - найбільш повно документоване, вільне і доступне інтегроване середовище розробки для Java. Eclipse також дуже простий у вивченні. Це робить Eclipse дуже привабливим IDE для розробки додатків під Android;

Компанія Google випустила плагін для Eclipse - Android Development Tools, який дозволяє створювати Android-проекти, компілювати їх, і, що дуже важливо, використовувати емулятор мобільного пристрою для запуску та налагодження додатків.

Плагін Android Development Tools для Eclipse автоматично створює необхідну структуру Android проекту і встановлює необхідні параметри налаштування компілятора.

Android SDK включає в себе різноманітні інструменти, бібліотеки та документацію, які допомагають розробляти мобільні додатки для платформи Android. У їх число входять:

Емулятор Android – віртуальний мобільний пристрій, який запускається на звичайному комп'ютері. Емулятор використовується для проектування, налагодження і тестування програм в реальному середовищі виконання Android.

Dalvik Debug Monitor Service (DDMS) - інтегрований з Dalvik, стандартною віртуальною машиною платформи Android, цей інструмент дозволяє керувати процесами на емуляторі або пристрої, а також допомагає у налагодженні додатків.

Операційна система Android є інноваційною, яка з кожним роком набирає все більшу і більшу популярність. Популярна операційна система для телефонів і інших гаджетів створена на базі Linux, яка була і залишилася конкурентом Windows від компанії Microsoft. ОС Android – це відкрита продукція, для якоъ є в наявності велика кількість різноманітних програм в безмежному безкоштовному доступі. До того ж ОС безперервно розвивається і поліпшується.

Володарі і творці ОС Android це компанія Open Headset Alliance, де налічується близько 80 різних фірм, навіть Google. 1-е дітище з даною системою на борту було виставлено на загальний огляд на початку 2008 року. Після чого послідували пропозиції від виробників мобільної техніки. Зараз Андроїд можливо зустріти не стільки на телефонах, але і на планшетних пристроях, а також і в фоторамках.

Основна частина програм для даної системи написана на мові Java. Завдяки опису операційної системи Android можна з'ясувати, як вона працює, які застосовуються двигуни і бібліотеки.

Починаючи з версії операційної системи під назвою Android 1.6 помітно перетворилася працездатність завантаження і покупки програм Android Market. Як і раніше, всі додатки поділяються на 2 масштабні групи Програми та Розваги, а далі на підкатегорії. У переліку підкатегорій показуються частоту завантаження контенту будь-якого з розділів. Найактуальніше нововведення - в опис програми або ж вставлені скріншоти, які дають можливість розцінити інтерфейс нового контенту. Це особливо необхідно при завантаженні комерційних програм, хоча крім того знадобиться і для приблизної оцінки дизайну нової заставки або гри, щоб не витрачати трафік і час даремно. Загалом, для вітчизняних користувачів, як і раніше доступні лише безкоштовні завантаження. Крім того спрацьовує система оцінки. Будь-який користувач, що завантажив програму, зможе розцінити її і кинути текстове пояснення для інших бажаючих завантажити.

Заключною версією на даний момент вважається Android 4.0 - Ice Cream Sandwich. У ній - поліпшені шрифти і анімація, комфортний багатозадачний інтерфейс, вдосконалена ревізія орфографії і потокове визначення голосу.

Безсумнівно, чудовий телефон з даною сильною операційною системою – це маленький комп'ютер, готовий виконати найскладніші завдання повсякденної необхідності.

Спочатку мова JAVA називався Oak («Дуб») розроблявся Джеймсом Гослінгом для програмування побутових електронних пристроїв. Згодом він був перейменований в Java і став використовуватися для написання клієнтських додатків і серверного програмного забезпечення. Названий на честь марки кави Java, яка, в свою чергу, отримала найменування однойменного острова (Ява), тому на офіційній емблемі мови зображена чашка з гарячою кавою. Існує й інша версія походження назви мови, пов'язана з алюзією на каву-машину як приклад побутового пристрою, для програмування якого спочатку мова створювалася.

Java є об'єктно-орієнтованою мовою програмування, розроблена компанією Sun Microsystems (в подальшому придбаною компанією Oracle). Програми Java можуть працювати на будь-якій віртуальної Java-машині незалежно від комп'ютерної архітектури. Дата офіційного випуску - 23 травня 1995 року.

Мова Java активно використовується для створення мобільних додатків під операційну систему Android. Розробку додатків можна вести в середовищі Android Studio, NetBeans, в середовищі Eclipse, використовуючи при цьому плагін Android Development Tools (ADT) або в IntelliJ IDEA. Версія JDK при цьому повинна бути 5.0 або вище.

8 грудня 2014 року Android Studio визнана компанією Google офіційним середовищем розробки під ОС Android.

У якості середовища розробки Android програми був обраний програмний продукт Android Studio. Він є інтегрованим середовищем розробки (IDE) для роботи з платформою Android, анонсована 16 травня 2013 року на конференції Google I / O.

IDE перебувала у вільному доступі починаючи з версії 0.1, опублікованій в травні 2013, а потім перейшла в стадію бета-тестування, починаючи з версії 0.8, яка була випущена в червні 2014 року. Перша стабільна версія 1.0 була випущена в грудні 2014 року, тоді ж припинилася підтримка плагіна Android Development Tools (ADT) для Eclipse.

Android Studio, заснована на програмному забезпеченні IntelliJ IDEA від компанії JetBrains, офіційний засіб розробки Android додатків [3]. Дане середовище розробки доступно для Windows, OS X і Linux. Основні особливості – реалізована можливість верстки в реальному часі, є безліч варіантів розмірів і дозволів екранів. Присутній розділ довідки. Вбудовані інструменти поліпшення якості додатків. Легкість в написанні коду. Готові шаблони основних макетів і компонентів Android. Є інструменти для відстеження ефективності рекламних оголошень.

3.2 Проектування основних модулів ігрового додатку

Для початку розглянемо деякі визначення, які в подальшому нам допоможуть не зупинятися на їх поясненнях [1]:

Кожен об'єкт в грі – це GameObject. Однак, об'єкти GameObject нічого не роблять самі по собі, їм необхідна спеціальна настройка, перш ніж вони стануть персонажами, предметами оточення або спеціальними ефектами. Якщо кожен об'єкт – це GameObject, то як ми зможемо розрізнити інтерактивні об'єкти і статичну кімнату? Що відрізняє ці об'єкти один від одного? Об'єкти GameObject є контейнерами. Порожня коробка, яка може містити всередині різні елементи, такі як гірська місцевість або автомобіль. Щоб дійсно зрозуміти GameObject, потрібно зрозуміти їх складові, який називаються компонентами (Components). Взалежності від того, який об'єкт створювати, можна буде додавати різні комбінації компонентів до об'єкту GameObject. GameObject як порожня тарілка, а компоненти як різні інгредієнти, які ми додаємо в неї. Так само можна створювати власні об'єкти за допомогою скриптів.

Game Assets – ігровий Ассет. Це свого роду набір будь-яких готових об'єктів, що відносяться до певного контенту, будь то модель персонажа, будинок, різні ефекти і багато іншого. Досить зручно працювати з GameObject в сцені, додаючи компоненти і змінюючи їх значення на потрібні. Однак це може створити ряд проблем в таких випадках, коли ми працюємо над створенням NPC, об'єктом або предметом, який багаторазово зустрічається в сцені. Можна просто скопіювати ці об'єкти для створення дублікатів, але всі вони будуть редагуватися незалежно один від одного. Зазвичай ми хочемо, щоб при зміні компонентів одного об'єкта, всі інші екземпляри так само мали ці компоненти, щоб не довелося редагувати всі окремо. На щастя, в Unity можна створювати префаб. Це особливий тип Ассет, що дозволяє зберігати весь GameObject зі всіма компонентами і значеннями властивостей. Префаб виступає в

ролі шаблону для створення екземплярів об'єкта в сцені. Будь-які зміни в префаб негайно відображаються і на всіх його примірниках, при цьому, можливо перевизначати компоненти і настройки для кожного екземпляра окремо.

Скрипти (Scripts) – невід'ємна складова кожної гри. Вони допомагають нам робити наші об'єкти на сцені такими, якими вони є в реальному житті, змінювати значення компонентів, завдяки їм об'єкти можуть реагувати на зовнішні впливи, визначати їх реакцію між іншими об'єктами і багато іншого. У Unity існують 3 мови на яких можливо писати скрипти – це C #, UnityScript (мова, розроблена спеціально для Unity заснована на JavaScript) і Boo.

Колайдери (Colliders) – це компоненти, які визначають форму об'єкта для визначення фізичних зіткнень. Вони невидимі і оточують об'єкт сіткою, яка може бути різної форми. Як примітивною, так і безпосередньої форми самого об'єкта. Існують 3 примітивів у коллайдера – Box Collider, Sphere Collider і Capsule Collider. Але якщо форма об'єкта не відповідає даним примітивам, можна використовувати Mesh Collider. Але вони використовують набагато більше навантаження на процесор, ніж примітивні типи. Тому використовувати їх потрібно економно, щоб підтримувати хорошу продуктивність. Однак хорошим правилом використання Mesh Colliders є застосування складових типів примітивних коллайдерів. Яким же чином у нас персонаж потрапляє в ігровий світ, як він пересувається, яким чином відбувається розвиток персонажа. Для цього відповімо спочатку на ці питання. Для цього важливо знати, яким чином працює Unity3D.

У Unity можна створювати сцени. Сцени містять об'єкти нашої гри. Вони можуть використовуватися для створення головного меню, окремих рівнів і для інших цілей. Можна рахувати кожен файл сцени окремим ігровим рівнем. У кожній сцені можна розмістити об'єкти оточення, загородження, декорації, по шматочках створюючи дизайн і саму гру. Для

нашого випадку буде шість сцен, кожна сцена буде окремим рівнем, в якому і будуть відбуватися дії, пов'язані з нашим персонажем. Так само на сцені міститися об'єкти, які ми додаємо, щоб втілити свою ідею.

Програми для Android складаються з компонентів, які система може запускати і керувати так, як їй необхідно. Для цього система повинна бути в змозі запустити процес для програми в якому знаходяться необхідні компоненти, і ініціалізувати потрібні їй об'єкти. Одним з компонентів Android - додатку є діяльність (activity).

Activity являє собою візуальний інтерфейс (окремий екран) для однієї дії, яку користувач може зробити.

Додаток може складатися з одного activity або з декількох. Це залежить від типу додатка і його дизайну. Одне activity може викликати інше.

Кожне activity задає вікно для відображення, яке, зазвичай, займає весь екран, але може бути менше і плавати поверх інших вікон. Activity може використовувати додаткові вікна, наприклад, спливаючий діалог, який вимагає проміжної відповіді користувача, або вікно, яке відображає користувачам важливу інформацію при виборі елемента, вартого особливої уваги.

Візуальний інтерфейс будується на основі ієрархії візуальних компонентів – об'єктів, похідних від базового класу View. Android має ряд готових до використання компонентів, включаючи кнопки, текстові поля, смуги прокрутки, меню, прапорці і багато іншого.

Перш ніж створювати додатки на мові Java, необхідно встановити інтегровану середу, JDK, JSDK, бібліотеки і зробити необхідні настройки.

Для роботи необхідно виконати наступні кроки:

- Завантажити і про інсталювати j2sdk1.5.0 (або іншу версію за адресою:

http://www.oracle.com/technetwork/java/archive-139210.html).

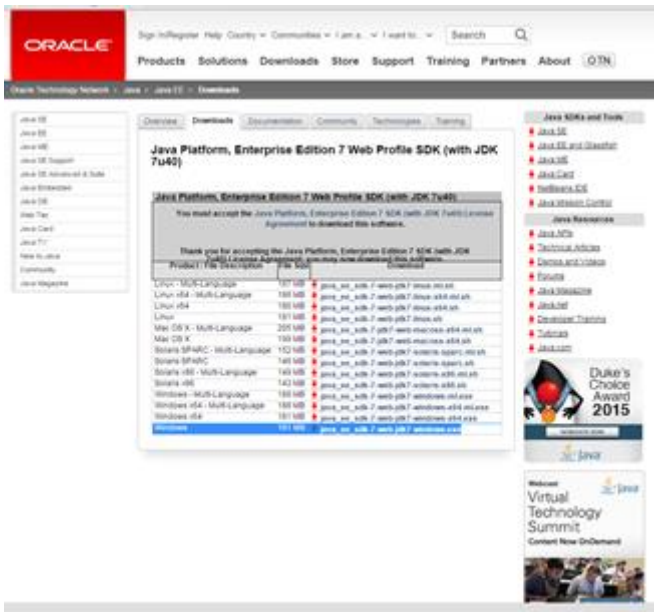


Рисунок 3.2 Налаштування середовища розробки

- Клацнути по exe-файлу.
- Перезавантажити комп'ютер.
- Прописати наступні змінні середовища. (Пуск> мій комп'ютер-> права кнопка миші-> властивості, вкладка: «Додатково - змінні середовища (оточення)»):

JAVA_HOME (рис. 3.3):

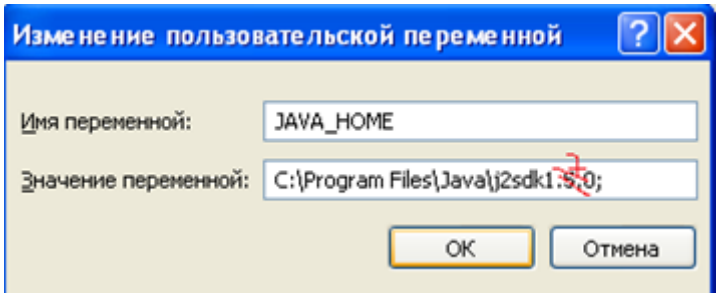


Рисунок 3.3 Змінна середовища JAVA_HOME

CLASSPATH (рис. 3.4):

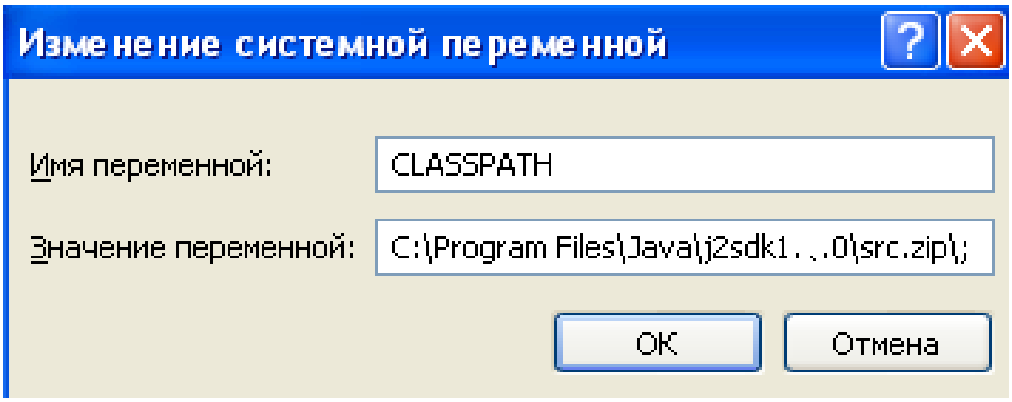


Рисунок 3.4 Змінна середовища CLASSPATH

відповідно до того файлового шляху, куди були розпаковані архіви.

- Створити папку в тому місці, де будуть зберігатися проекти (наприклад, в корені, на диску "D", під назвою 'MyWebProjects') (рис. 3.5):

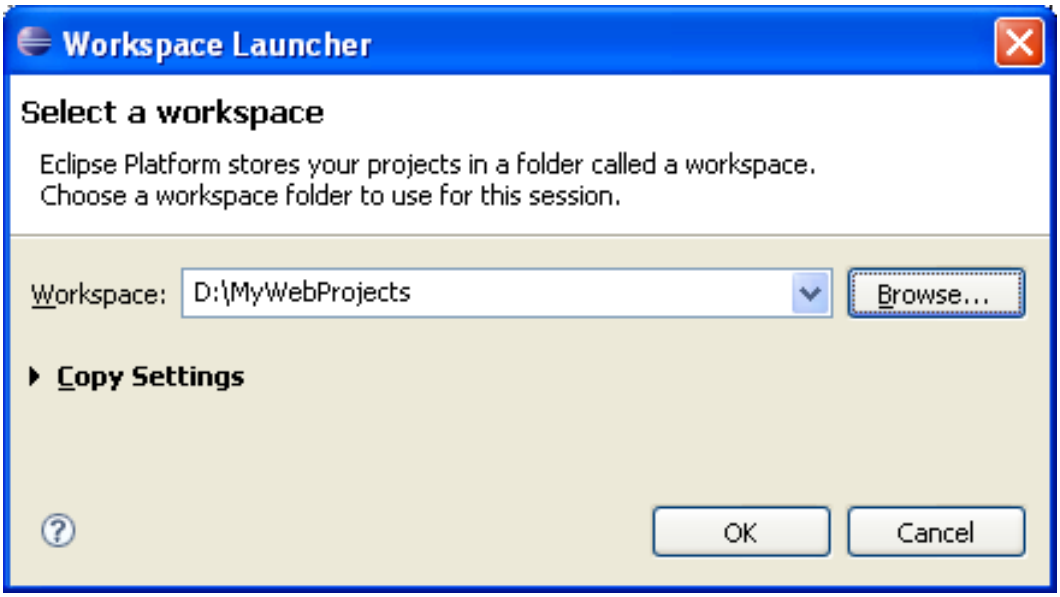


Рисунок 3.5 Вибір Workspace

- Запустити Eclipse, клікнувши по файлу з ярликом Екліпса - синім кружком. Надалі є можливість, наприклад, закріпити його в меню «пуск». Як Workspace у вікні за запитом потрібно вибрати заготовлену папку.
- Через деякий час з'явиться Welcome-сторінка Eclipse з ярликами (рис. 3.6):

Рисунок 3.6 Welcome-сторінка Eclipse

- Потрібно клацнути лівою кнопкою миші на ярлику 'Workbench', і відкриється перспектива 'Resource', яку Eclipse відображає за замовчуванням (рис. 3.7):

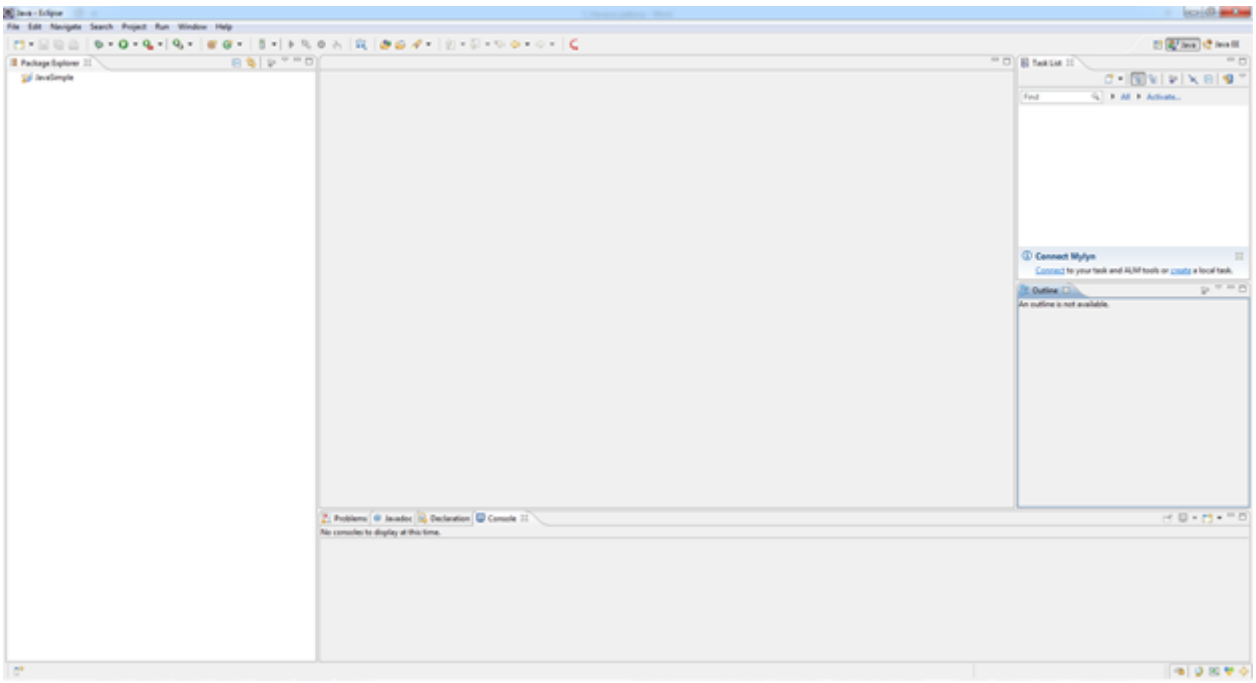


Рисунок 3.7 Перспектива 'Resource'

- Для того щоб створювати Java-додатки, потрібно створити проект. Для цього натисніть на вкладку File-> New-> Java Project і дайте ім'я проекту, наприклад, 'JavaSimple' або 'MySimpleProject' (рис. 3.8):

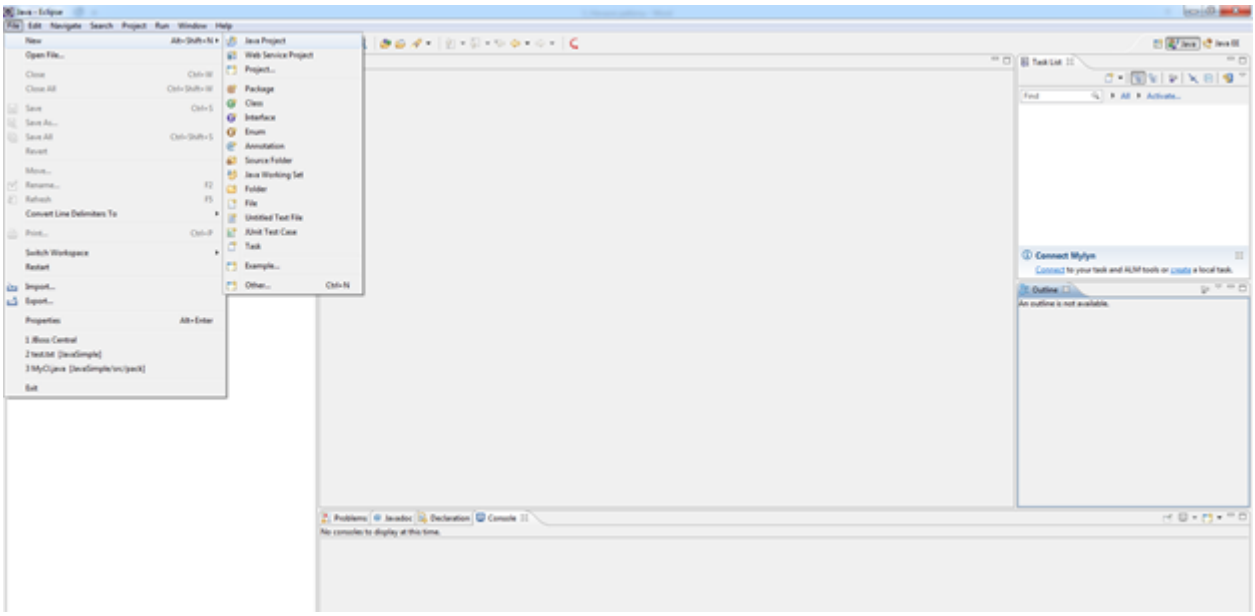


Рисунок 3.8 Створення проекту

Тепер знайдіть папку src (source) і створіть свій пакет для додатків. Дайте йому назву, наприклад, 'pack'.

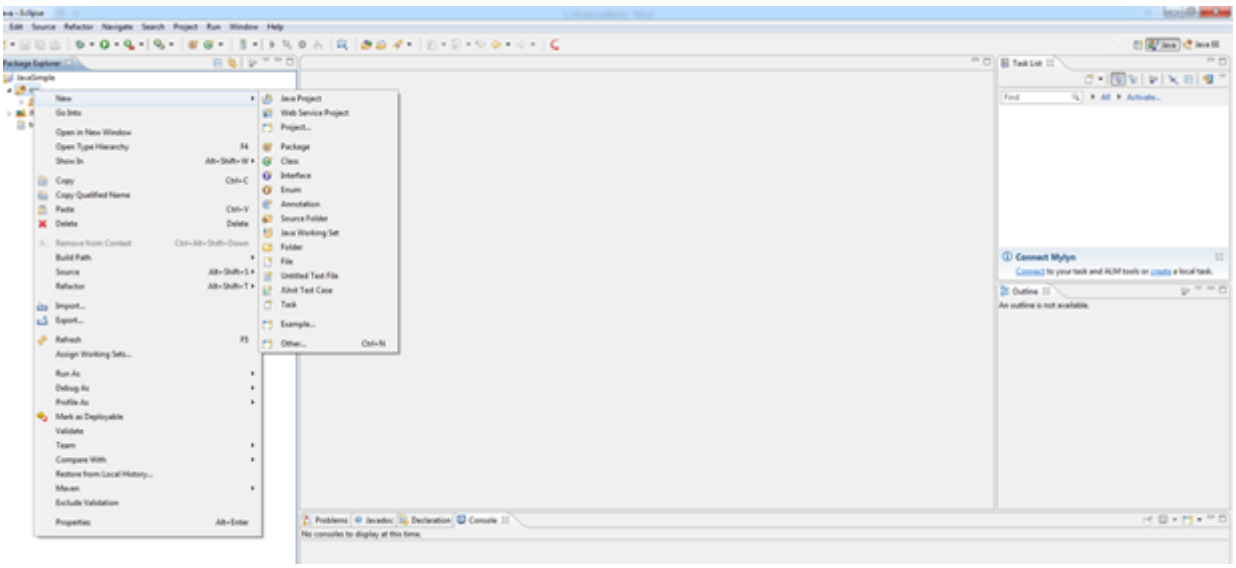


Рисунок 3.9 Створення проекту

Створіть тепер перший клас (права клавіша миші по пакету - вибираємо New / Class).

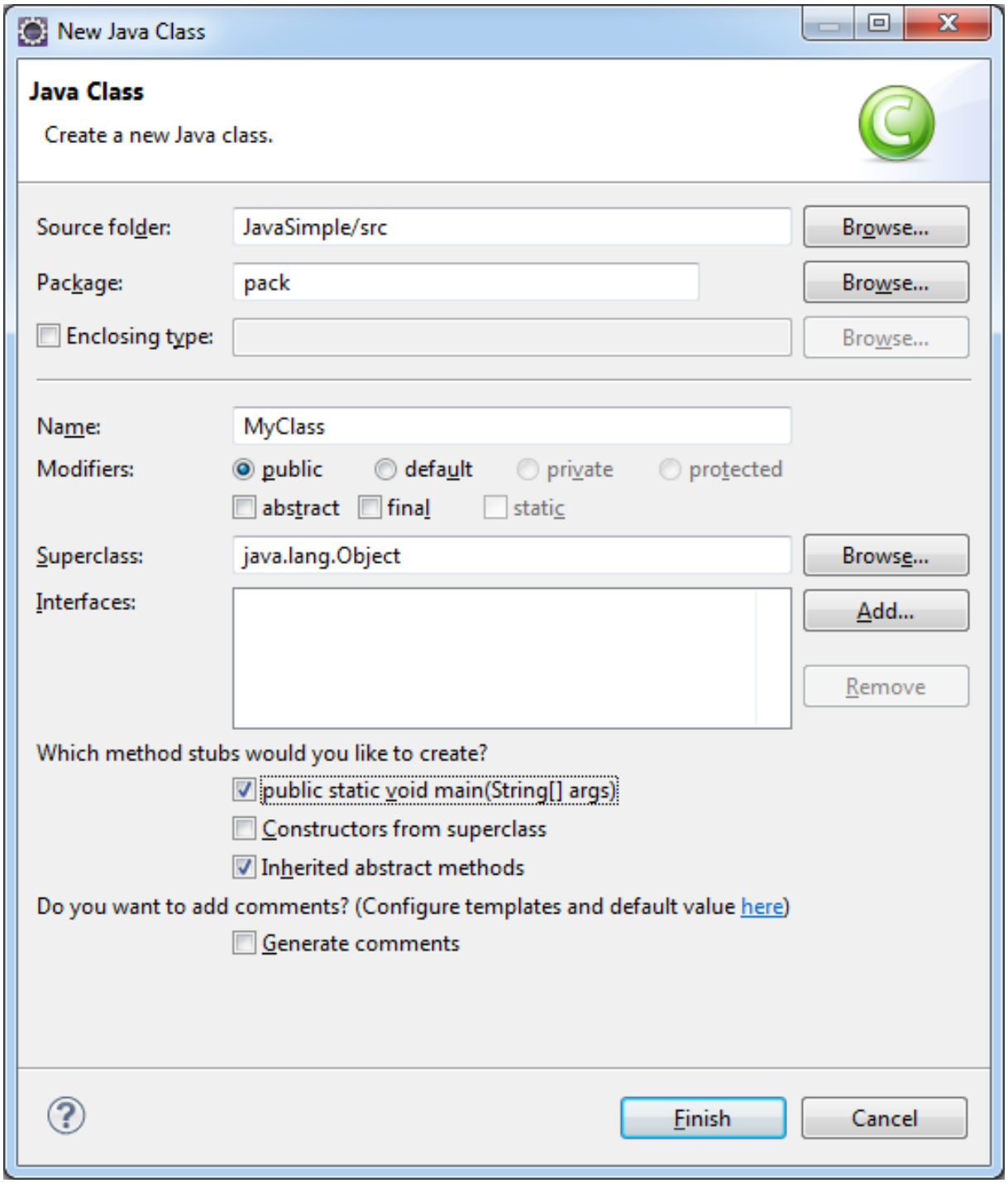


Рисунок 3.10 Створення класу

Натиснимо Finish і отримаємо заготовку під клас.

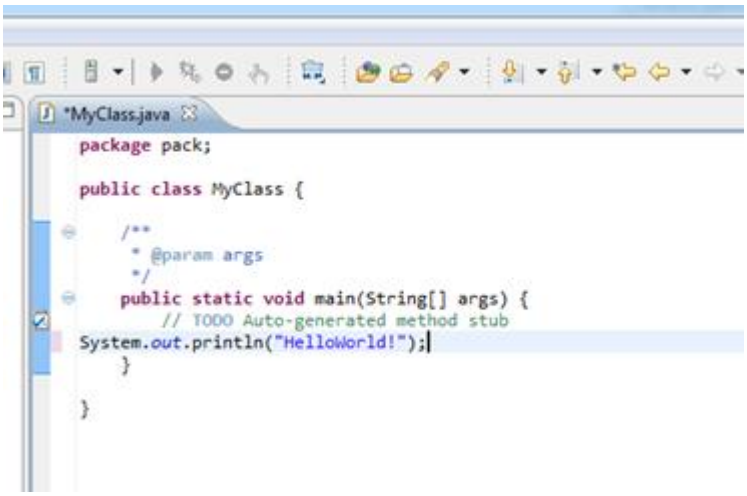


Рисунок 3.11 Формування додатку

Натисніть на зелений трикутник в горизонтальній верхньої панелі:

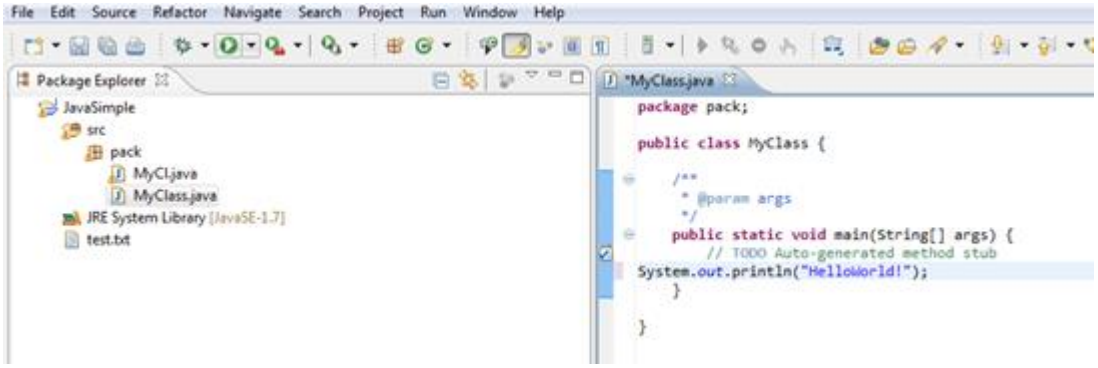


Рисунок 3.12 Формування додатку

Виберіть Java Application:

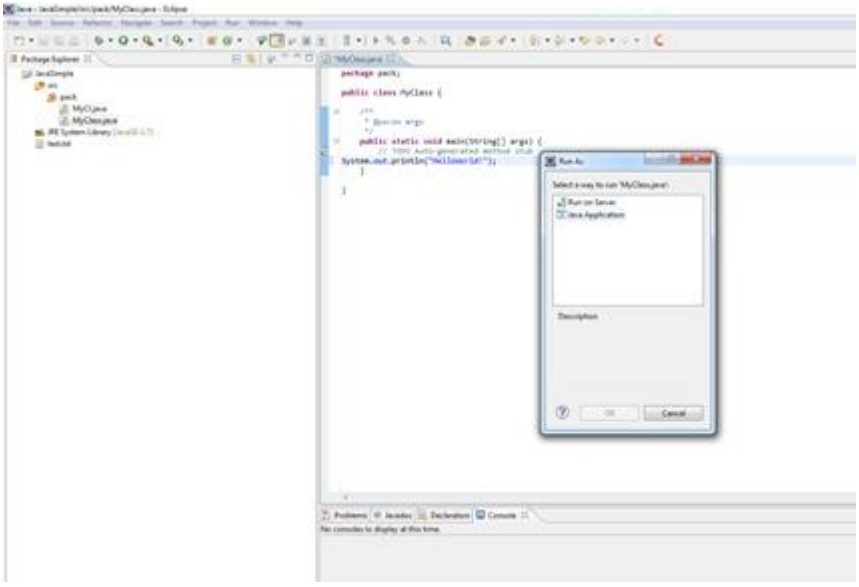


Рисунок 3.13 Формування додатку

І внизу в консолі відіб'ється ваш текст:

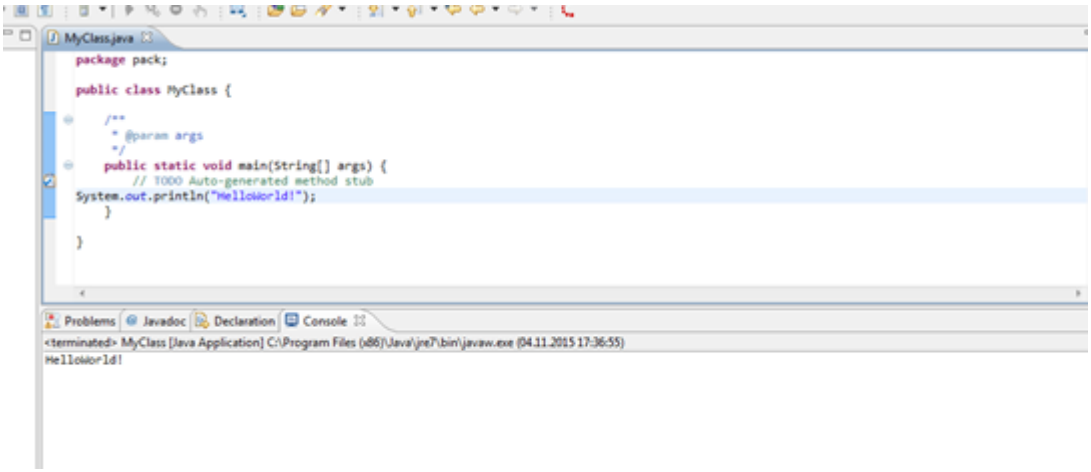


Рисунок 3.14 Формування додатку

Протестуйте його (наприклад, виведіть напис або підрахуйте арифметичний вираз).

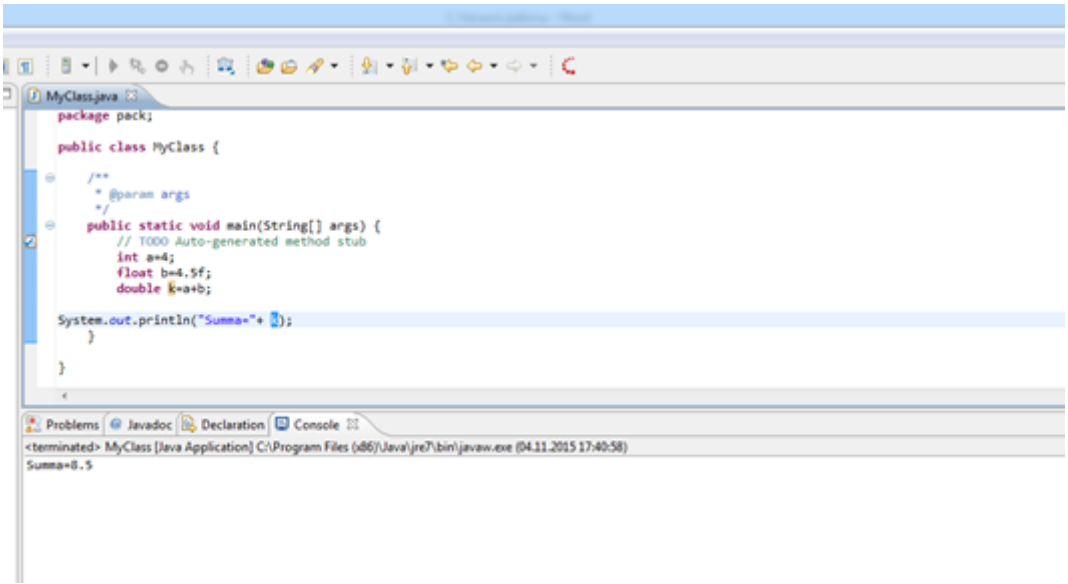


Рисунок 3.15 Формування додатку

Здійснюємо проектування всіх основних компонентів мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

Розглянемо види об'єктів, які будуть потрібні для нашої розробки:

1) Персонаж

Це об'єкт, яким управлятиме користувач. Він необхідний для того, щоб ми могли переміщатися по ігровому світу і взаємодіяти з іншими об'єктами.

2) Об'єкт NPC

Це об'єкт, який необхідний для перешкоджень та нагород.

3) Об'єкт Canvas

Даний об'єкт пов'язаний з інтерфейсом користувача. У ньому містяться панелі, клавіші, текст, які будуть створюватися з після різних дій персонажа, керованого користувачем.

1. Персонаж

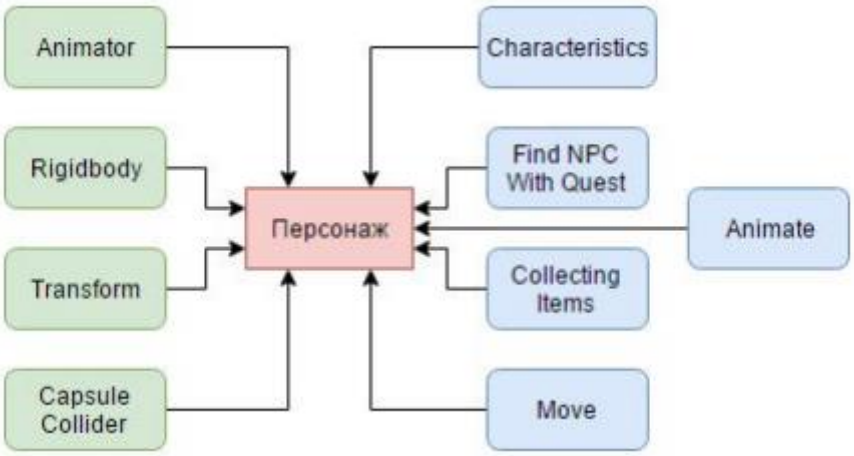


Рисунок 3.16 Компоненти об'єкта «Персонаж»

Як ми бачимо з рис. 3.16, на нашому персонажі прикріплено досить багато різних компонентів.

Сині компоненти – це скрипти.

1) Компонент Animator

Цей компонент дозволяє нам контролювати анімацію персонажа. Він приймає контролер, в якому задана послідовність програвання анімації. Завдяки цьому компоненту при знаходженні в нерухомому стані, програється анімація «простою», при русі вперед, вліво, вправо, програється анімація «їзди».

2) Компонент Rigidbody

Це основний компонент, що включає фізичне поводження для об'єкта. З прикріпленим Rigidbody, об'єкт відразу ж почне реагувати на гравітацію. В даному компоненті, можливо вказувати масу об'єкта, змусити активувати кінематичний рух. Кінематичний рух – це такий рух, при якому фізичний движок не управляє рухом персонажа.

3) Компонент Transform

Даний компонент існує у всіх об'єктів на сцені при їх створенні. Він відповідає за стан відповідних об'єктів на сцені. Безпосередньо їх розташування, поворот, а також масштаб.

4) Компонент Capsule Collider

З назви бачимо, що це коллайдер. Даний коллайдер складається з двох півсфер, з'єднаних між собою циліндром.

5) Компонент Characteristics

Даний компонент і наступні, які будуть розглянуті на поточному об'єкті – це скрипти, кожен з яких відповідає за своє дія.

6) Компонент Find NPC with Quest

В даному скрипті відбувається вся логіка прийняття, виконання та завершення завдань. У цьому компоненті відбувається відстеження натискання правою клавішею миші на NPC.

7) Компонент Animate

Даний скрипт допомагає міняти анімації в залежності від дій персонажа, будь то ходьба, або стояння не місці.

8) Компонент Collecting Items

Цей скрипт реалізує логіку одного з варіантів завдань. А саме – збір тих чи інших предметів. Як говорилося вище в скрипті Find NPC with Quest, при прийнятті завдання, у нас створюються необхідні об'єкти, які ми можемо зібрати. В даному скрипті реалізовано відстеження натискання на даний об'єкт, процес його збору (він займає 1 секунду), а також процес відродження заново даних об'єктів (воно займає 1 хвилину).

9) Компонент Move

Даний скрипт відповідає за логіку пересування персонажа. Тобто при натисканні на клавіші WASD, персонаж починав рухатися у відповідних напрямках: вперед, назад, вліво і вправо. Увесь цей рух відбувається з однаковою швидкістю, яку ми заздалегідь поставили. Переміщення відбувається за допомогою зміни властивості компонента Transform.position на величину, що дорівнює добутку вектора напрямку, швидкості і часу між зміни кадрами.

2. Об'єкт NPC

На рис. 3.17 показані компоненти об'єкта «NPC».

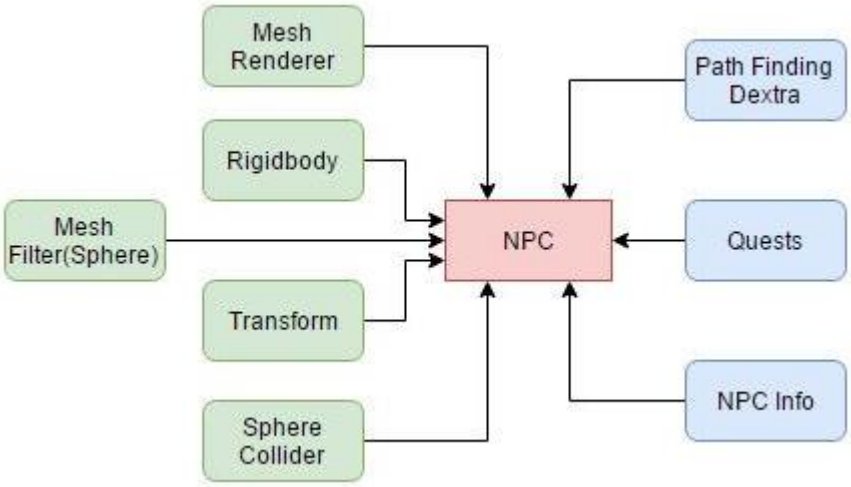


Рисунок 3.17 Компоненти об'єкта «NPC»

1) Компоненти Mesh Filter (Sphere) і Mesh Renderer

Два даних компонента взаємопов'язані між собою. Mesh Filter зчитує геометрію нашого Accet і передає її Mesh Renderer, щоб той її намалював. При створенні об'єктів, які є примітивами, ці 2 компонента автоматично вже є у них.

2) Компонент Rigidbody

Як говорилося вище, це компонент, що включає фізичну поведінку для об'єкта.

3) Компонент Transform

Стандартний компонент, що відслідковує положення об'єкта в просторі координат.

4) Компонент Sphere Collider

Базовий коллайдер в формі сфери.

5) Компонент NPC Info

Даний скрипт містить інформацію про ім'я, id, початкових координатах NPC. Id ми задаємо кожному NPC самі, але ім'я ми зчитуємо з файлу.

6) Компонент Quests

В даному скрипті ми отримуємо всю інформацію по всіх завданнях, які мають всі NPC. Вся ця інформація надходить з файлів, які знаходяться в папці з Ассет, в певному стилі. На рис. 3.18 показано, яким чином знаходяться файли для зчитування в системі.

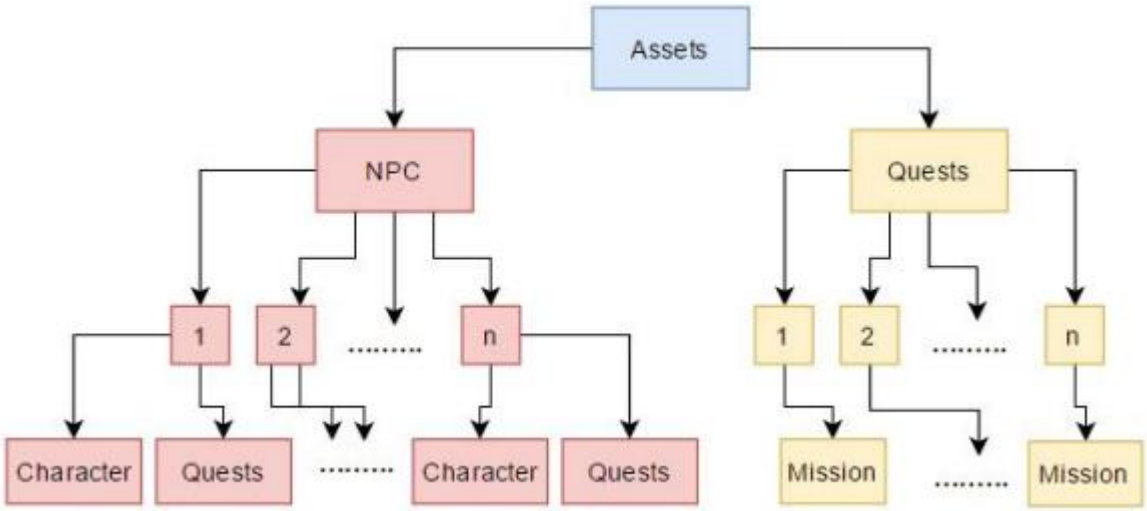


Рисунок 3.18 Структура файлів в системі

Спираючись на схему, можна наочно зрозуміти яким чином знаходяться файли в системі.

3.3 Ігровий цикл

Серцем гри є ігровий цикл (рис. 3.19), який здійснює почерговий виклик всіх модулів в правильній послідовності і синхронізацію їх між собою. Наш ігровий цикл буде реалізований в скрипті.

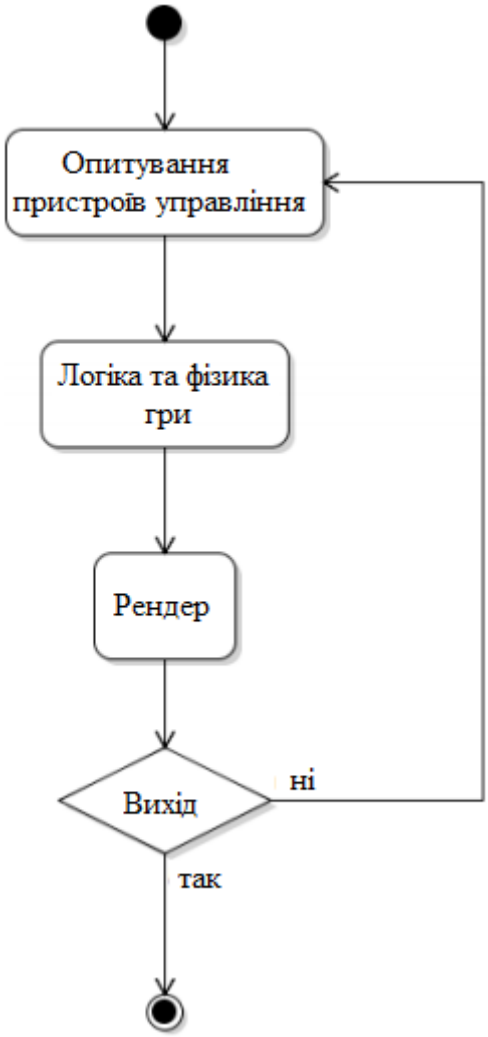


Рисунок 3.19 Простий ігровий цикл

Ігровий цикл гри, що розробляється буде складатися з:

- опитування стану зовнішніх пристроїв управління;
- зміна положень ігрових об'єктів;
- підрахунок очок;

- оновлення координат фізичних об'єктів сцени;
- оновлення елементів меню і інтерфейсу.

Використання класичного ігрового циклу в Unity неможливо з-за особливості архітектури движка, яка не передбачає ручне управління всіма викликами основних модулів. Кожен об'єкт в Unity має в скрипті два обов'язкових методи. Перший метод Start () для ініціалізації об'єкта і використовується один раз при додаванні об'єкта на сцену. Другий метод називається Update (), Unity викликає його кожен раз і у кожного об'єкта, перед тим як оновити всю сцену. Ми можемо використовувати метод Update для імітації ігрового циклу, написавши свій скрипт і підключивши його до камери сцени (рис. 3.20).

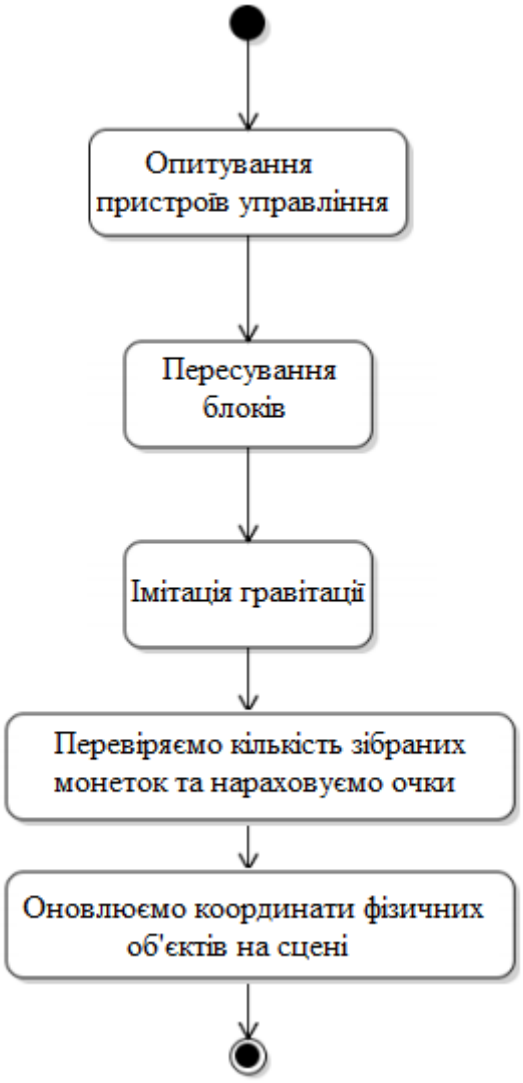


Рисунок 3.20 Ігровий цикл гри

Метод Update скрипту буде виконувати наступні дії:

- здійснювати запит у Unity про стан зовнішніх пристроїв управління;
- пересувати блок;
- пересувати блоки, що знаходяться в стані "польоту" і ще не впали на поле;
- обновляти координати об'єктів на сцені.

3.4 Система меню

Система меню у грі розроблена максимально зрозумілою. У лівому верхньому куті буде показано кількість очок, що зібрав гравець під час проходження рівня. Верхня частина розбита на три вкладення: налаштування, гараж та магазин. У правому нижньому куті винесено активну кнопку натискання на яку призводить до початку гри.

У лівому нижньому куті є вкладення «друзі» при натисканні на нього є можливість переглянути всіх гравців процесу . Також нижня панель розбита на окремі вкладення, такі як статистика перемог, нагороди, рейтинг та подарунки.

РОЗДІЛ 4
ЕТАПИ РЕАЛІЗАЦІЇ МУЛЬТИПЛЕЄРНОЇ ГРИ FUNSHOT MULTIPLAYER

4.1 Створення головного меню

Головне вікно програми наведено на рис. 4.1.

Рисунок 4.1 Головне вікно програми

Систему меню розписано у попередньому розділі. Модулі активації задано як:

```
GameObject:
  m_ObjectHideFlags: 0
  m_PrefabParentObject: {fileID: 0}
  m_PrefabInternal: {fileID: 100100000}
  serializedVersion: 4
  m_Component:
  - 224: {fileID: 22400000}
  - 223: {fileID: 22300000}
  - 114: {fileID: 11400002}
```

```
  - 114: {fileID: 11400000}
```

Перегляд за рівнями гри виконано як:

```
m_Name: LookUpAndDownTouchpad
  m_TagString: Untagged
  m_Icon: {fileID: 0}
  m_NavMeshLayer: 0
  m_StaticEditorFlags: 0
  m_IsActive: 0
--- !u!1 &100004
GameObject:
  m_ObjectHideFlags: 0
  m_PrefabParentObject: {fileID: 0}
  m_PrefabInternal: {fileID: 100100000}
  serializedVersion: 4
  m_Component:
  - 4: {fileID: 400000}
  - 114: {fileID: 11400012}
  m_Layer: 0
  m_Name: TiltSteerInput
  m_TagString: Untagged
  m_Icon: {fileID: 0}
  m_NavMeshLayer: 0
  m_StaticEditorFlags: 0
  m_IsActive: 0
--- !u!1 &100006
```

Кожна гра побудована на композиції елементів.

4.2 Створення елементів гри

Переміщення об’єкта відбувається за допомогою процедури:

```
MonoBehaviour:
  m_ObjectHideFlags: 1
  m_PrefabParentObject: {fileID: 0}
  m_PrefabInternal: {fileID: 100100000}
  m_GameObject: {fileID: 100002}
  m_Enabled: 1
  m_EditorHideFlags: 0
  m_Script: {fileID: 11500000, guid: 1caf40fc8bebb6b43b2550c05ca791d6, type:
3}
  m_Name:
  m_EditorClassIdentifier:
  axesToUse: 0
  controlStyle: 2
  horizontalAxisName: Mouse X
  verticalAxisName: Mouse Y
  Xsensitivity: 1
  Ysensitivity: 1
```

4.3 Створення логіки та фізики додатка

Для всіх взаємодіючих на сцені об'єктів, таких як блоки і платформи, нам потрібно зберігати інформацію з координатами на сцені, їх розміри, напрям руху, статус і так далі. Для цього ми створимо клас Party і прив'яжемо всі об'єкти до нього. Діаграму можна побачити на рисунку 4.2.

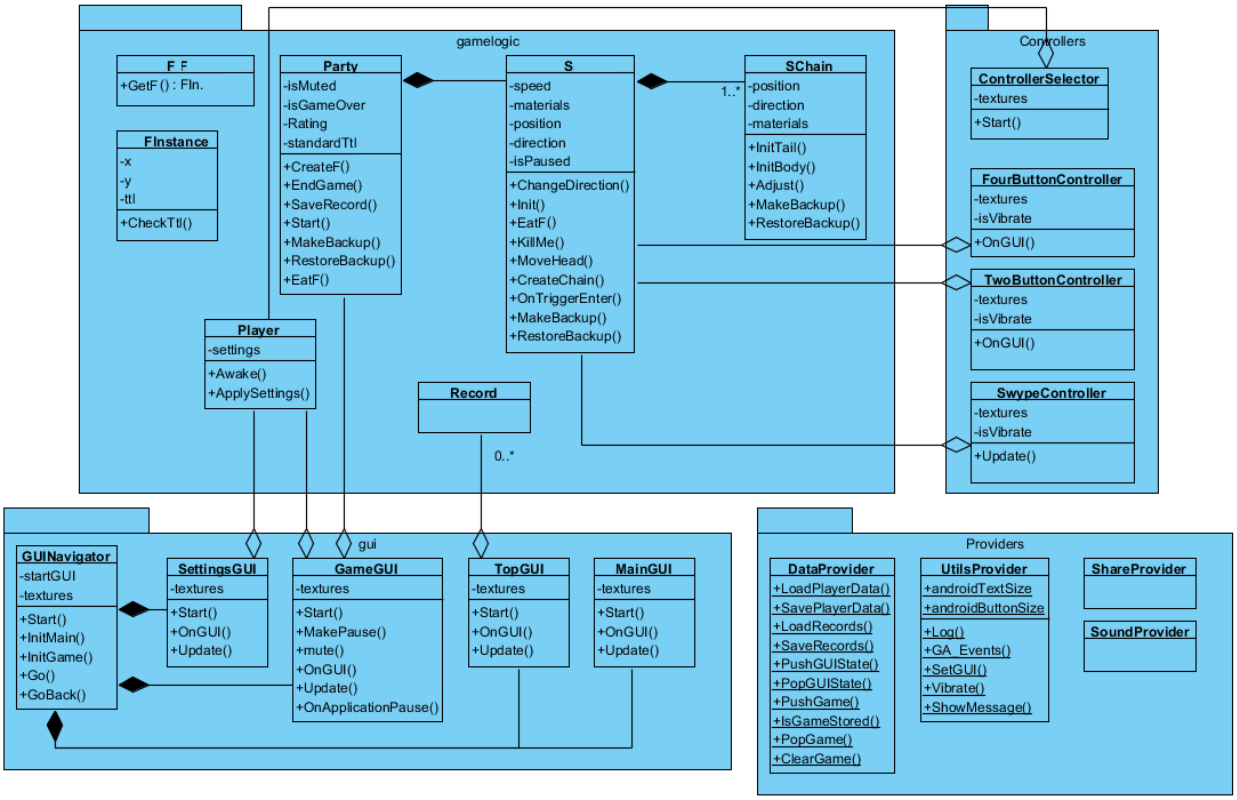


Рисунок 4.2 Діаграма класів

Клас Gui буде такі методи:

- start () починає гру;

- pause () запиняє гру;

- mute () формує поле;

- ongui () переміщає блоки;

- add () додає об'єкт в пул;

- solver () перевіряє об'єкт на зіткнення;

- checkEqualBoxes () перевіряє однакові бокси на зіткнення і якщо воно сталося видаляємо зі сцени і нараховуємо гравцеві бали.

- update () оновлює фізичні координати об'єктів на сцені, перемальовує їх за один прохід;

- reset () повертає координати блоків в первісний стан для перезапуску рівня.

Після того як завантажений рівень, Unity передає управління до сцени скрипту. Він в свою чергу зчитує всі наявні на сцені об'єкти з тегами блоків і платформ, ініціалізує їх і поміщає в загальний пул.

Далі Unity буде кожен раз перед оновленням сцени передавати управління методу Update () в якому ми в свою чергу будемо опитувати пристрої управління, переміщати блоки, оновлювати елементи інтерфейсу.

Для того щоб зібрати сцени, текстури, спрайт і код програми воєдино потрібно використовувати Build settings у візуальному редакторі Unity.

Слід перенести використовувані сцени з проекту у вікно Scenes to build, вибрати цільову платформу, і вибрати Build and Run.

4.4 Тестування розробленого додатка

Проведемо тестування розробленої гри. Головним елементом гри є автомобіль-корабель (рис. 4.3), який курсує по ігровому полю та ловить супротивників, обстрілюючи їх та отримуючи за це нарахування балів та монет (рис. 4.4).

Рисунок 4.3 Головний елемент – автомобіль-корабель

Поле гри є яскравим з використанням різних елементів у просторі (рис. 4.4).

Рисунок 4.4 Обстріл супротивників

Рисунок 4.5 Поле гри з використанням різних елементів у просторі

Під час тестування збоїв та недоліків не виявлено.

ВИСНОВКИ

У рамках даної дипломної роботи здійснено розробку мультиплеєрної грури FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

Проаналізовано проблему впливу відеоігор на розвиток дітей. Правильно підібрані відеоігри розвивають у дітей мислення, увагу і логіку, пам'ять і здатність приймати рішення. Управління відеогрою покращує координацію рухів дитини. Навчальні ігри допомагають підготувати його до школи. Групові відеоігри вчать терпінню і взаємодії в групі. Деякі ігри розвивають творчу уяву.

Розроблено локацію, яка складається з різних видів ландшафту. В розроблено додатку локація представлена набором різних моделей, які розміщені по всьому простору. Також в роботі реалізовано управлінням персонажем. Завдяки даному контролеру можливе пересування персонажа з використанням навігації.

Описано програмні засоби, котрі було застосовано для розробки програмного забезпечення. Визначено, що оптимальним рішенням є використання офіційного середовища розробки Android Studio та Unity. У даній роботі були розглянуті основні аспекти розробки ігрових додатків. Движок Unity показав себе як зручний, простий в освоєнні та професійно виконаний інструмент для створення ігор.

Якість та працездатність мультиплеєрної грури FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android забезпечуэ проведене тестування. Тестові випадки створені для перевірки основного функціоналу на базі діаграми використання застосунку.

Застосунок буде ефективним для людей, що цікавляться іграми подібного роду. Також, рекомендується використовувати застосунок як

доповнення до лекцій з «Розробка додатків для смартфонів» у навчальних закладах, для кращого засвоєння матеріалу.

Робота пройшла апробацію на наступних конференціях:

1. Вимоги та характеристики до розробки мультиплєєрно гри на платформії UNITY 3D//Розробка ігор. Збірник тез. – К.: ДУТ, 2021. — С. 105 – 106., м. Київ, 12.02.2020

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Miles R. C# Programming Yellow Book Ed.8.1, 2016 – 216 p.
2. Хокинг Д. Unity в действии. СПб.: Питер, 2016. – 336 с.
3. *Стиллмен А. Изучаем C#*,2012. – 704с.
4. Албахари Б. C# 6.0. Полное описание языка / Б. Албахари. Дж.Албахари. 2016 – 1040с.
5. Об’єктно-орієнтоване програмування – режим доступу: http://programming.in.ua/programming/basisprogramming/25-oop.html (16.10.2020)
6. developer.microsoft.com – режим доступу: https://msdn.microsoft.com/en-us/ (18.10.2020)
7. Игровые объекты (Game Object) [Электронный ресурс] / Unity 3D Game Development. Статьи и уроки по разработке игр под Unity 3D. – 2016. – Режим доступа: http ://unity-dev. ru/basics_gam e_objects. (18.10.2020)
8. Создание сцен [Электронный ресурс] / Unity 3D Game Development. Статьи и уроки по разработке игр под Unity 3D. – 2016. – Режим доступа: http://unity-dev.ru/building_scenes. (18.10.2020)
9. Игровой Ассет. [Электронный ресурс] // Википедия: свободная энцикл., 2017. – Режим доступа: https://ru.wikipedia.org/wiki/Игровой_ассет/ (дата обращения 18.10.2020).
10. Руководство пользователя Unity. – [Электронный ресурс]. – Режим доступа: https://docs.unity3d.com/ (дата обращения: 18.10.2020).
11. Русскоязычное сообщество Unity3d.ru. – [Электронный ресурс]. – Режим доступа: http://unity3d.ru/ (дата обращения: 18.10.2020).
12. Официальный сайт Unity3d. – [Электронный ресурс]. –Режим доступа: https://unity3d.com/ru/ (дата обращения: 18.10.2020).
13. Джозеф Х. Unity в действии. Мультиплатформенная разработка на C#. Спб: Питер, 2016. – 336 с.

14. Global Positioning System [Электронный ресурс]. – Режим доступа: http://www.navcen.uscg.gov. – Загл. с экрана.

15. Enabling GPS programatically in Android. [Электронный ресурс]. – Режим доступа: http://stackoverflow.com/questions/5715257/enabling-gps-programatically-in-android. – Загл. с экрана.

16. General Information on GPS [Электронный ресурс]. – Режим доступа: http://www.navcen.uscg.gov. – Загл. с экрана.

17. Android - Architecture / Tutorialspoint [Офиц. сайт]. [Электронный ресурс]. – Режим доступа: http://www.tutorialspoint.com/android/android_architecture.htm 26.10.2020

18. Introduction to Glide, Image Loader Library for Android, recommended by Google / The Cheese Factory: Blog [Офиц. сайт]. [Электронный ресурс]. – Режим доступа: http://inthecheesefactory.com/blog/get-to-know-glide-recommended-bygoogle/en. 26.10.2020

19. Material design - Introduction / Google [Офиц. сайт]. [Электронный ресурс]. – Режим доступа: https://www.google.com/design/spec/materialdesign/introduction.html. 26.10.2020

20. Android Wear by Google / Android [Офиц. сайт]. [Электронный ресурс]. – Режим доступа: https://www.android.com/wear/ 26.10.2020

21. Google Play Hits 1 Million Apps [Электронный ресурс]. – Режим доступа: http://mashable.com/2013/07/24/googleplay-1-million/ 26.10.2020

22. Android App Stats [Электронный ресурс]. – Режим доступа: http://www.androlib.com/appstats.aspx 26.10.2020

23. Google: 3 Billion Android Apps Installed; Downloads Up 50 Percent From Last Quarter [Электронный ресурс]. – Режим доступа: http://techcrunch.com/2011/04/14/google-3-billion-android-appsinstalled-up-50-percent-from-last-quarter/ 26.10.2020

24. Smartphone OS Market Share, Q1 2015 [Электронный ресурс]. – Режим доступа: http://www.idc.com/prodserv/smartphone-os-market-share.jsp 26.10.2020

25. Apps for Programming on Android [Электронный ресурс]. – Режим доступа: http://android.appstorm.net/roundups/developer/15-apps-for-programming-onandroid/ 26.10.2020

26. The Perfect Platform for Game Developers: Android [Электронный ресурс]. – Режим доступа: http://www.developer.com/ws/android/client/the-perfect-platform-for-gamedevelopers-android.html 16.05.2020

27. Introduction to Android [Электронный ресурс]. – Режим доступа: https://developer.android.com/guide/index.html 21.05.2020

28. Jake Wharton, [Web – resource], Exploring RxJava 2 for Android – access [Электронный ресурс]. – Режим доступа: https://youtu.be/htIXKI5gOQU GOTO 2016 open access.

29. David Karnok, [Web – resource] RxJava backpressure – access. [Электронный ресурс]. – Режим доступа: https://github.com/ReactiveX/RxJava/wiki/Backpressure open access, June 27, 2016.

30. T. Nield [e-book], Learning RxJava, Packt Publishing, 2017, 400 с.

31. App Components [Электронный ресурс]. – Режим доступа: https://developer.android.com/guide/components/index.html

32. AndroidSecurityBulletin – January 2017 [Електронний ресурс]. – Режим доступу : https://source.android.com/security/bulletin /2017-01-01.html – Назва з екрану

33. Мобильные вирусы и угрозы. Статистика [Електронний ресурс]. – Режим доступу : http://www.kaspersky.ua/internet-security-center/threats/mobile/

34. Karpachev I., Kazymyr V. Functional security in an ANDROID mobile architecture / I. Karpachev, V. Kazymyr // Вісник Чернігівського державного технологічного університету. 2015. № 1 (77).

35. Saad M. H., Serageldin A., Salama G. I. Android spyware disease and medication / M. H. Saad, A. Serageldin, G. I. Salama // Information Security and Cyber Forensics (InfoSec), 2015 Second International Conferenceon. – IEEE, 2015. C. 118–125.

36. Mark L. Murphy. The Busy Coder's Guideto Android Development / Mark L. Murphy // Commonsware, LLC. 2013. 443 p.

37. Understanding security on Android. [Электронный ресурс]. – Режим доступа: http://www.ibm.com/developerworks/library/xandroidsecurity.

38. Fedler R. Android OS security: risks and limitations / R. Fedler, C. Banse, C. Kraub, V. Fuesing. 2012. P. 19–27.

39. Mobile application security on android, context on android security / J. Burns // BlackHat. 2009. 27 p.

40. Ratazzietal P. A systematic security evaluation of android's multiuser framework / P. Ratazzietal // Proc. IEEE MoST, 2014. P. 1–10.

41. Fedler R. Effectiveness of malware protection on android and evaluation of android antivirus apps / R. Fedler, J. Schutte, M. Kulicke // Applied and integrated security. 2014. P. 7–13, 26–32.

42. Мелешко О. О. Способи захисту інформації з обмеженим доступом в мобільних пристроях від витоку / О. О. Мелешко, О. С. Болотнікова // Сучасний захист інформації. 2016. № 1.

43. Войтович О. П. Дослідження інцидентів безпеки в ОС Android / О. П. Войтович, М. В. Гурський // Тези доповідей XLV Науково-технічної конф. Вінницького національного технічного університету / ФІТКІ – Вінниця, 2016. С.55-60.

ДОДАТОК

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА МУЛЬТИПЛЕЄРНОЇ ГРИ FUNSHOT MULTIPLAYER В ЖАНРІ ШУТЕР З ВИКОРИСТАННЯМ ІГРОВОГО ДВИГУНА UNITY ДЛЯ ПЛАТФОРМИ ANDROID

Виконав студент 4 курсу
групи ПД-42
Осадчий Богдан Олександрович
Керівник роботи
к.т.н., доцент Негоденко Олена Василівна

Київ – 2021

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- Комп'ютерна гра складається з різного роду компонентів, які розробляються багатьма програмістами, дизайнерами, сценаристами і іншими людьми, кожен з яких відіграє свою роль у проекті. Багатьох, комп'ютерні ігри просто рятують, тому що допомагають пережити тяжкі часи або стрес, коли поруч нікого з близьких людей немає.

- **Метою дипломної роботи** є створення мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

- **Об’єкт дослідження** виступає процес проектування та розробки мультиплеєрної гри у жанрі Shooter.

- **Предмет дослідження** ігровий двигун UNITY та платформа Android.

АНАЛОГИ

3

ТЕХНІЧНІ ЗАВДАННЯ

- *1.* розкрити теоретичних аспектів створення якісної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android;
- *2.* сформувати проблематику дослідження та постановка завдань;
- *3.* розробити основні модулі мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android та опис структурних компонентів;
- *4.* провести тестування створеної мультиплеєрної гри FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android.

4

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

Операційна система Android – портативна (мережева) операційна система для комунікаторів, планшетних комп'ютерів, цифрових програвачів, наручних годинників.

Java Development Kit (JDK) - безкоштовно розповсюджуваний корпорацією Oracle Corporation комплект розробника додатків мовою Java, що включає в себе стандартні бібліотеки класів Java, приклади, документацію, різні утиліти і виконавчу систему Java (JRE).

Eclipse - найбільш повно документоване, вільне і доступне інтегроване середовище розробки для Java. Eclipse також дуже простий у вивченні.

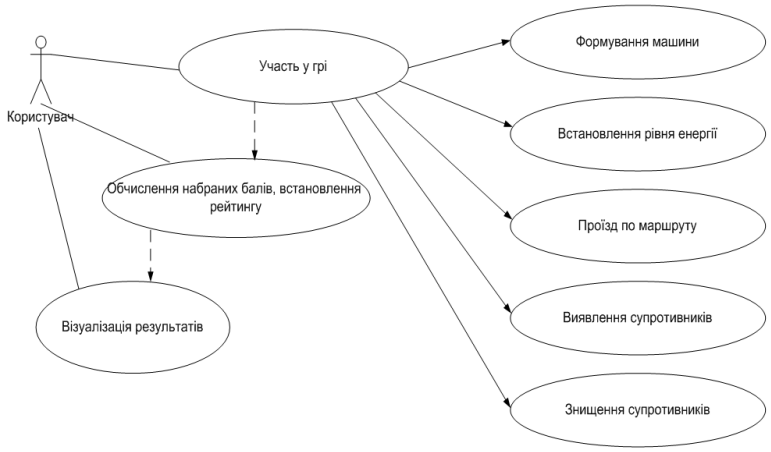
Unity — багатоплатформовий інструмент для розробки відеоігор і застосунків, і рушій, на якому вони працюють. Створені за допомогою Unity програми працюють на настільних комп'ютерних системах, мобільних пристроях та гральних консолях у дво- та тривимірній графіці, та на пристроях віртуальної чи доповненої реальності.

5

МЕТОДИ ТА КЛАСИ ПРОГРАМИ

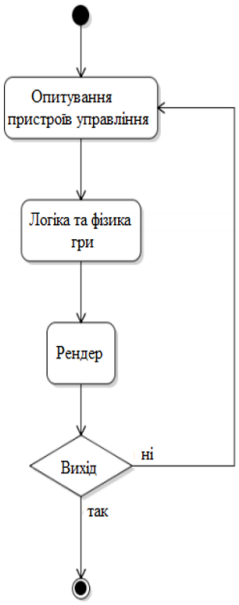
– Діаграма використання

6

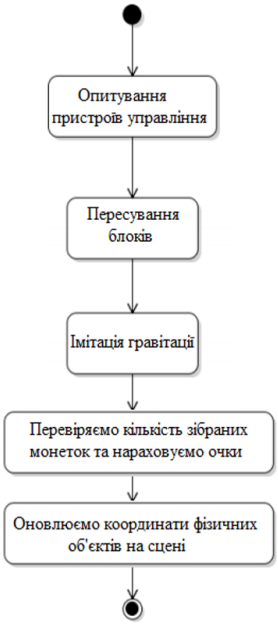


Діаграма прецедентів

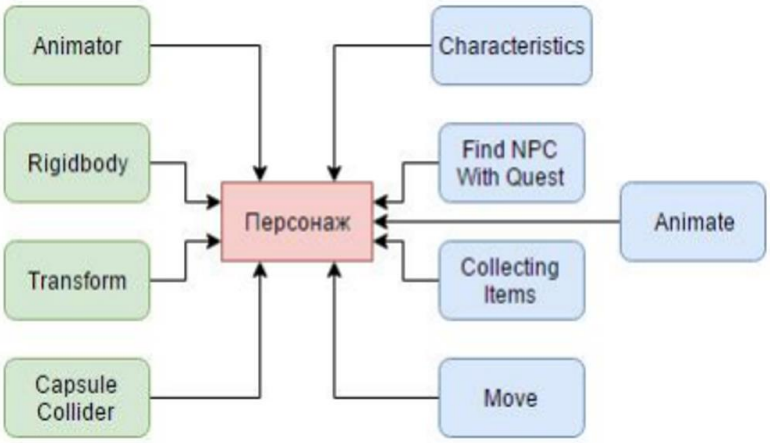
7



Простий ігровий цикл



8



Компоненти об'єкта «Персонаж»

9

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Осадчий Б.О. Вимоги та характеристики до розробки мультиплєєрно гри на платформії UNITY 3D
//Розробка ігор. Збірник тез. – К.: ДУТ, 2021. — С. 105 – 106.

10

ВИСНОВКИ

- При виконанні даної дипломної роботи було спроектовано і розроблено мультиплеєрну гру FunShot multiplayer в жанрі шутер з використанням ігрового двигуна UNITY для платформи Android. Цей додаток не є закінченим продуктом, так як вимагає наповнення професійно виконаним графічним контентом, але при відповідному доопрацюванні може зайняти свою нішу на ринку ігор.
- У даній роботі розроблена локація, яка складається з різних видів ландшафту. Локація – це місце, в якому персонажі можуть виконувати ті чи інші дії. Локація може бути представлена набором різних моделей, які розміщені по всьому простору. Також в роботі реалізовано управлінням персонажем. Завдяки даному контролеру можливе пересування персонажа з використанням навігації.
- Система меню у грі розроблена максимально зрозумілою. У лівому верхньому куті буде показано кількість очок, що зібрав гравець під час проходження рівня. Верхня частина розбита на три вкладення: налаштування, гараж та магазин. У правому нижньому куті винесено активну кнопку натискання на яку призводить до початку гри.

11

ДЯКУЮ ЗА УВАГУ!