МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ІГРОВОГО ЗАСТОСУНКУ ЗАСОБАМИ UNITY»**

Виконав: студент 4 курсу, групи 6.1219-1пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
М.О. Шкуропєй
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, к.ф.-м.н. Кривохата А.Г.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри комп'ютерних наук, доцент, к.т.н. Решевська К.С.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ 07 ” 02 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Шкуропєю Максиму Олеговичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка ігрового застосунку засобами Unity

керівник роботи Кривохата Анастасія Григорівна, к.ф.-м.н.
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 26 « січня 2023 року № 102-с

2. Строк подання студентом роботи 07.06.2023 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Розробка ігрового застосунку засобами Unity.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою докладу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Гребенюк С. М., завідувач кафедри фундаментальної та прикладної математики		
2	Гребенюк С. М., завідувач кафедри фундаментальної та прикладної математики		
3	Гребенюк С. М., завідувач кафедри фундаментальної та прикладної математики		

7. Дата видачі завдання 07.02.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	08.02.2023	
2.	Збір вихідних даних.	15.02.2023	
3.	Обробка методичних та теоретичних	08.03.2023	
	джерел.		
4.	Розробка першого та другого розділу.	12.04.2023	
5.	Розробка третього розділу.	15.05.2023	
6.	Оформлення та нормоконтроль	01.06.2023	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	22.06.2023	

Студент _____ М.О. Шкуропєй
(підпис) (ініціали та прізвище)

Керівник роботи _____ А.Г. Кривохата
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка ігрового застосунку засобами Unity»: 42 с., 12 рис., 12 джерел, 1 додаток.

ІНТЕГРОВАНЕ СЕРЕДОВИЩЕ РОЗРОБКИ MICROSOFT VISUAL STUDIO, МОВА ПРОГРАМУВАННЯ C#, ПЛАТФОРМА ANDROID, РУШІЙ РОЗРОБКИ UNITY, ADOBE ILUSTRATOR, ANDROID STUDIO.

Об’єкт дослідження – процес розробки Android застосунку, що реалізує гру.

Мета роботи – розробка ігрового застосунку за допомогою Unity.

Мета дослідження – розробити застосунок, що реалізує гру для Android пристроїв, із застосуванням засобів Unity.

Метод дослідження – аналітичний.

У кваліфікаційній роботі бакалавра розглядаються основні концепції розробки ігрового застосунку з використанням засобів Unity. Розглядаються основні етапи розробки, такі як проєктування, реалізація геймплею, робота з графікою, інтеграція з системою Android та багато іншого.

У процесі розробки ігрового застосунку з використанням Unity використовуються такі інструменти, як Microsoft Visual Studio для програмування на мові C#, яка є основною мовою програмування в Unity.

Для розробки графічних елементів, ілюстрацій та інтерфейсу можна використовувати Adobe Illustrator або інші графічні редактори. Unity має підтримку для імпорту графіки з різних форматів, що дозволяє використовувати створені ілюстрації у розробці ігрового застосунку.

SUMMARY

Bachelor’s qualifying paper «Development of a game application using Unity»: 42 pages, 12 figures, 12 references, 1 supplement.

MICROSOFT VISUAL STUDIO INTEGRATED DEVELOPMENT ENVIRONMENT, C# PROGRAMMING LANGUAGE, ANDROID PLATFORM, UNITY DEVELOPMENT ENGINE, ADOBE ILLUSTRATOR, ANDROID STUDIO.

The object of the study is the process of developing an Android application that implements a game.

The aim of the study is to develop a game application using Unity.

The purpose of the study is to develop an application that implements a game for Android devices using Unity tools.

The method of research is analytical.

The bachelor's thesis discusses the basic concepts of developing a game application using Unity tools. The main stages of development are considered, such as design, gameplay implementation, graphics, integration with the Android system, and much more.

In the process of developing a game application using Unity, tools such as Microsoft Visual Studio for programming in C#, which is the main programming language in Unity, are used.

Adobe Illustrator or other graphic editors can be used to develop graphic elements, illustrations, and the interface. Unity has support for importing graphics from various formats, which allows you to use the created illustrations in the development of a game application.

ЗМІСТ

ВСТУП

У сучасному світі важко уявити собі людину без різних портативних пристроїв, таких як мобільні телефони, планшети, смартфони та інші мультимедійні пристрої. Ці пристрої завжди знаходяться під рукою і виконують не тільки функцію засобу спілкування, але і мають багато корисних можливостей, таких як калькулятор, органайзер, конвертер і календар, а також служать годинником.

Сучасні смартфони з новими мобільними ігровими платформами навіть можуть конкурувати зі стандартними портативними ігровими системами, такими як Nintendo DS або Playstation Portable.

Структура смартфона в основному складається з кількох окремих компонентів, таких як пам'ять, процесор, який відповідає за обчислення, пам'ять для зберігання даних та радіо-модуль, який містить передавач і приймач і забезпечує зв'язок. Однак, найбільш цікавим елементом є операційна система, яка встановлюється на внутрішню пам'ять пристрою. Від версії та типу операційної системи залежать всі основні можливості пристрою. Існують різні операційні системи для смартфонів та персональних комп'ютерів, про які буде детальніше розповідатися надалі.

Оскільки мобільні продажі в усьому світі зростають, також зростає попит на різні додатки для них. Кожна поважаюча себе компанія прагне мати, щонайменше, один мобільний додаток, щоб бути його клієнтом і мати «завжди під рукою». А існування деяких компаній взагалі складно уявити без мобільних та спеціалізованих програм, з допомогою яких можна, наприклад, управляти базами даних, або контролювати стан свого продукту на ринку в будь-який момент часу.

На жаль, на сьогоднішній день не існує будь-яких конкретних стандартних інструментів розробки мобільних додатків. Кожен виробник намагається зробити операційну систему на пристрої унікальною і такою, що

запам’ятовується користувачеві, і як наслідок виникають проблеми сумісності між різними додатками на різних операційних системах.

Комп’ютери стають все більш «особистими» з можливістю доступу до них в будь-який час і з будь-якого місця. На передньому краї цього процесу стоять мобільні пристрої, які трансформуються в обчислювальні платформи. Мобільні телефони вже давно використовуються не тільки для розмов – вони можуть бути використані протягом певного періоду часу для передачі даних і відео. Мобільні пристрої стали виконувати широкий спектр обчислювальних загальних завдань, ці пристрої можуть стати новим поколінням персональних комп’ютерів (ПК). Крім того, навіть очікувалося, що деякі виробники традиційних моделей ПК – зокрема, ASUS, HP і Dell – зроблять пристрої, багато конструктивних параметрів яких будуть базуватися на OS Android.

Незабаром в IT-індустрії, як очікується, буде швидко збільшуватися в розмірі і обсязі програмне забезпечення для мобільних пристроїв.

Ця нова тенденція відкриває доступ до мобільних пристроїв для традиційних мов програмування, так що область застосування мобільних додатків і їх частка на ринку ростуть.

Таким чином, актуальність теми представленого проєкту очевидна. Найближчим майбутнім розвиток і підтримка додатків, заснованих на операційній системі Android, буде найбільш затребуваним на ринку програмного забезпечення.

Одним з видів таких додатків, є комп’ютерні ігри. Навчальні та розвиваючі ігри – ігри, в яких ігровий процес є основою розвитку і вдосконалення різних навичок. Концепція розвиваючих ігор пояснюється головним чином періодами життя дитини. Діти грають в розвиваючі ігри, готують своє власне мислення, винахідливість, уяву, творчі здібності.

1 АНАЛІЗ ІНФОРМАЦІЇ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Комп’ютерні ігри

У сучасному світі комп'ютерні технології широко застосовуються в різних сферах нашого життя. Процес розробки комп'ютерних програм, як простих, так і складних, для різних галузей знань постійно розвивається. Залежно від віку дитини, комп'ютерні програми можуть виконувати різні функції, такі як опонент у грі, оповідач, репетитор або екзаменатор. Вже існують комп'ютерні програми, спрямовані на розвиток різних психічних функцій у дітей, таких як візуальне і слухове сприйняття, увага, пам'ять, мовні і логічні навички тощо. Ці програми успішно використовуються дітьми старшого дошкільного віку та молодшого шкільного віку. Зараз настільні комп'ютери поступово замінюються мобільними пристроями, такими як планшетні ПК та смартфони, що дозволяє зручно працювати зі всією необхідною інформацією.

Гра - це пізнавальна діяльність, яка представляє собою особливу практичну форму розуміння природи соціальної дійсності, що оточує дитину. Через свої особливості, гра дозволяє дитині вперше включитися в абстрактне мислення.

Комп'ютерні ігри стали новим видом розвиваючого навчання. Завдяки комп'ютерним технологіям відкриваються нові можливості для використання педагогічних методів:

- завжди можна індивідуально підібрати матеріал різної складності для дитини, що відповідає його поточним здібностям та навчальним цілям;
- допомагають виявити труднощі, пов'язані з розвитком дитини, які можуть бути важко помітити в традиційному навчанні;
- для того, щоб сформувати розуміння дитиною процесу отримання

власних навичок;

- складне програмне забезпечення є надзвичайно простим в використанні;
- з точки зору фахівця, комп'ютеризація багатьох раніше успішно використовуваних методів надає можливість переглянути свою роботу з нової перспективи, переосмислити методичні прийоми та збагатити свої знання і навички;
- робота на комп’ютері створює для дитини більш комфортні умови для успішного виконання вправ;
- комп’ютерні технології забезпечують захоплюючі заняття для дитини в формі експериментування, моделювання, порівняння;
- дитина навчиться правильно говорити, намагатиметься виправити побачену помилку, шукатиме методи самоконтролю, орієнтуючись на привабливу графіку;
- діти менше втомлюються, більше здатні працювати;
- дивлячись на екран, дитина сама бачить результат своєї роботи.

Завдяки грі, на основі якої побудована програма, використання комп’ютерних програм підвищує мотивацію. Дитина отримує схвалення і похвалу не тільки від дорослого, а й від самого комп'ютера. Однак життєво важливо підходити до використання комп’ютерних ігор зі здоровим глуздом і творчо підходити до їх вибору. Велика частина гри спрямована на розвиток психічних процесів, таких як пам’ять, увага, уява, сприйняття, мислення, розвиток координації очей і рук. Перед тим, як запропонувати гру для дітей, необхідно ознайомитися з її метою і змістом [2].

1.2 Огляд існуючих мобільних платформ

Оперативна система (ОС) є основною характеристикою, яка відрізняє смартфон від звичайного мобільного телефону. Операційна система часто є

ключовою при виборі пристрою або моделі телефону.

Найбільш поширені операційні системи для смартфонів і платформ наведені нижче.

Symbian OS – до кінця 2010 року Symbian OS була найпопулярнішою операційною системою для смартфонів. Лише одна платформа на базі цієї ОС, Series 60, залишилася з 2010 року. У деяких моделях Samsung і Nokia вона використовується в основному.

BlackBerry OS (RIM) – є популярною операційною системою на пристроях, особливо в Сполучених Штатах. Це пов'язано з тим, що деякі країни обмежують використання смартфонів, оскільки алгоритм AES шифрує вхідну та вихідну інформацію. Це приваблює спецслужби тих країн до цієї системи.

Windows Mobile і **Windows CE** – це компактні операційні системи від Microsoft, які почали випускатися з 1996 року. Вони займали найбільший сегмент ринку операційних систем для смартфонів до 2010 року. Однак, в даний час відбувається поетапна відмова від підтримки та розвитку цих систем.

Windows Phone 7 – розробка від Microsoft, радикально відрізняється від Windows Mobile.

Palm OS – була однією з популярних платформ, але на даний момент мобільні телефони на базі Palm OS є рідкістю. Останній смартфон, що працював під управлінням цієї операційної системи, був випущений в кінці 2007 року і називався Palm Centro.

Linux – операційна система, яка не дуже популярна на мобільних пристроях, але традиційно вважається перспективним розвитком. Смартфони на базі Linux широко використовуються в Азії.

Bada – остання мобільна платформа від Samsung. S8500 Wave був першим телефоном на новій платформі.

Android – операційна система на базі Linux, розроблена для використання на смартфонах, планшетних ПК, електронних книгах, цифрових

плеєрах, годинниках і нетбуках. Android був створений Android Inc., але потім його придбав Google. Після цього Google заснував альянс Open Handset Alliance (OHA), щоб підтримувати та розвивати платформу. Android дозволяє розробляти програми на основі Java, які взаємодіють із пристроями за допомогою бібліотек, розроблених Google. Крім того, Android Native Development Kit дозволяє системі використовувати компоненти та бібліотеки програм, написаних на мові C, серед інших мов.

IOS – є мобільною операційною системою, розробленою і створеною американською компанією Apple. Вона також називавалась iPhone OS до 24 червня 2010 року. У 2007 році вона була випущена для пристроїв iPhone і iPod Touch. Пізніше вона була розширена для iPad і Apple TV. iOS доступний лише для пристроїв Apple, на відміну від Windows Phone і Android від Google..

Windows Phone 8 – друге покоління телефонної операційної системи від Microsoft Windows. Є прототипом інтерфейсу, відомого як Metro (або Modern UI). Windows Phone 8 використовує нову архітектуру Windows NT, яка використовується в операційних системах Microsoft. Пристрої, що працюють під управлінням Windows Phone 7.X, не можуть оновити операційну систему до Windows Phone 8, а нові додатки, створені для Windows Phone 8 не можуть працювати на Windows Phone7.X [1].

В даний час Android розвивається в геометричній прогресії: щороку число користувачів цієї операційної системи постійно зростає. Згідно з останнім звітом компанії Canalys, провідного аналітика індустрії високих технологій, операційна система Android займає 69,2% світового ринку мобільних пристроїв. Звичайно, цей факт спонукає багатьох розробників створювати мобільні додатки, спеціально для Android. Можливо, на сьогоднішній день вона є найпопулярнішою і цікавою системою. Розробники дають користувачам унікальну можливість – встановити набір вільного програмного забезпечення, можна створювати програми для системи і продавати їх в спеціалізованому інтернет-магазині.

1.3 Опис предметної області

Розроблена Google операційна система Android базується на платформі Linux для мобільних пристроїв. Вона була створена Open Handset Alliance (OHA) і дозволяє створювати програми на основі мови програмування Java, які використовують бібліотеки, розроблені Google, для керування пристроями. Крім того, можна створювати програми на мові C та інших мовах програмування за допомогою Android Native Development Kit. Версія 1.5 Android, також відома як Cupcake, була випущена 30 квітня 2009 року і містила деякі значні оновлення. Вони включали підтримку запису та перегляду відео в режимі камери, підтримку Bluetooth A2DP (яка дозволяє прослуховувати аудіо за допомогою Bluetooth) і можливість автоматично підключати гарнітуру Bluetooth.

T-Mobile G1 був першим пристроєм під управлінням Android, розробленим HTC. Він вийшов 23 вересня 2008 року. Після цього низка інших виробників смартфонів повідомила про свої наміри випустити смартфони на базі Android.

Пристрої на базі платформи Android відрізняються від подібних продуктів кількома важливими перевагами:

- відкритість – завдяки своїй відкритості Android дозволяє розробникам отримати доступ до основних функцій мобільного пристрою за допомогою звичайного API виклику;
- руйнування кордонів – відкриває нові можливості, коли дані з телефону, такі як контактні дані та географічні координати, поєднуються з інформацією з Інтернету;
- рівність додатків – програми на телефоні та програмне забезпечення від сторонніх виробників мають рівні можливості на платформі Android. Можна навіть змінити програму набору номерів або заставку, щоб відповідати вашим уподобанням;
- швидко і легко розробляти програми – набір засобів розробки

програмного забезпечення, або SDK, містить всі інструменти, необхідні для створення та запуску додатків для Android, що робить розробку швидкою та простою. SDK включає інструменти для налагодження коду та імітатор справжнього пристрою, що дозволяє тестувати програми.

Гнучкість платформи Android має свою ціну, оскільки компанії часто вирішують розробляти власні інтерфейси для користувача і постійно випускають нові версії операційної системи. Це може призвести до швидкого застаріння пристроїв, які випущені лише кілька місяців тому, оскільки оператори і виробники не завжди надають оновлення програмного забезпечення, які дозволяють користувачам отримати нові функції Android. Наприклад, хоча платформа Android базується на Java, переваги і можливості операційної системи Linux не використовуються повною мірою. Також в ній відсутні популярні графічні інструменти (наприклад, Toolkit) та бібліотеки (наприклад, Qt або GTK), що ускладнює перенесення багатьох додатків з повною версією для комп'ютерів на мобільну платформу. Крім того, відомо, що Google може видаляти додатки з пристроїв користувачів, якщо вони порушують умови використання. Ці фактори обмежують деякі можливості і роблять деякі аспекти розвитку додатків на Android менш доступними для розробників.

Експерти та аналітики прогнозують, що Google Android має чудові комерційні перспективи на IT-ринку. Наявність програмного забезпечення з відкритим вихідним кодом не є чимось новим на сьогоднішній день, але Android успішно витісняє відомих лідерів у IT-сфері. Це створює здорову конкуренцію, яка може сприяти відновленню ринку та створенню нових інноваційних товарів.

Більшість коду платформи Android ліцензується Apache 2. Можна вільно використовувати вихідний код за допомогою цієї безкоштовної та відкритої ліцензії для створення власних систем. Це дозволяє розробникам і компаніям використовувати Android для створення інноваційних продуктів і рішень,

розширюючи функціональність платформи та адаптуючи її до своїх потреб. Ліцензія сприяє розширенню екосистеми Android і розробці різноманітних користувачів-додатків і сервісів. Проте, система, щоб бути сумісною з Android, повинна бути сумісною з програмами для Android – процес сертифікації базової сумісності зі сторонніми додатками, які створюються сторонніми розробниками. Сумісні системи можуть вливатися в екосистему Android, включаючи Android Market [3].

2 ОПЕРАЦІЙНА СИСТЕМА ANDROID

2.1 Архітектура Android

З програмістської точки зору, Android є платформою, яка дозволяє розробникам писати код на мові Java і не турбуватися про деталі ядра. Android має багато цінних функцій. По-перше, він надає широкий спектр API, які дозволяють створювати різноманітні програми та замінювати компоненти, які надаються як платформою, так і сторонніми виробниками. По-друге, запуск програм на Android залежить від віртуальної машини Dalvik. Крім того, Android підтримує різноманітні мультимедійні формати, такі як Ogg Vorbis, MP3, MPEG-4, H.264 та PNG, а також надає API для доступу до камери, GPS, компаса, акселерометра, сенсорного екрана, джойстика та клавіатури. Існує навіть спеціальний API для відтворення фонових звукових ефектів, що корисно для нас в розробці ігор. Не всі Android-пристрої мають всі ці можливості – є апаратний підрозділ. Звичайно, у Android не вичерпується список можливостей згаданих тут. Проте, для розробки ігор вони є найбільш важливими. Android-архітектура формується з безлічі компонентів. Кожен компонент заснований на елементах нижнього рівня [7].

Драйвери апаратних компонентів системи надаються ядром Linux. Крім того, ядро відповідає за пам’ять, мережеву підтримку, управління процесами та інші функції. Розробка та функціонування програм Android здійснюється середовищем виконання Android, побудованим навколо ядра. Кожна програма використовує віртуальну машину Dalvik для виконання окремого процесу.

Програми Dalvik, які працюють на Android, можуть запускати за допомогою форми байткоду DEX (Dalvik Executable). Інструмент DX, який доступний у програмному наборі засобів розробки програмного забезпечення (SDK), використовується для перетворення файлів Java, які містять класи, у формат DEX. Завдяки високому ступеню стиснення, поділу на таблиці та

об’єднанню кількох CLASS-файлів формат DEX потребує значно менше пам’яті, ніж класичний формат CLASS. Це покращує продуктивність виконання програм і оптимізує використання пам’яті мобільних пристроїв.

Віртуальна машина Dalvik взаємодіє з бібліотеками ядра, які надають базові функціональні можливості для Java-програм. Ці бібліотеки мають значний, але не повний набір класів, доступних через Java SE.

До версії Android 2.2 (Froyo) весь код інтерпретувався. У Froyo було введено JIT-компілятор, який може компілювати частини байткоду на льоту в машинний код. Це суттєво покращує продуктивність додатків, які потребують більше обчислювальних ресурсів. JIT-компілятор може використовувати особливості процесорів, які спеціалізовані на складних обчисленнях, наприклад, операціях з плаваючою комою. Крім того, він включає власний збирач сміття Dalvik (Garbage Collector, GC), який працює за принципом "повідомити і забрати". Хоча це може ставити розробників у виклик, якщо його ретельно вивчити, він може бути ефективно використаний при розробці ігор. Кожен додаток, що працює в інстанції віртуальної машини Dalvik, має доступ до пам'яті розміром від 16 до 24 МБ RAM. Це слід враховувати при роботі з графічними і звуковими ресурсами. Крім основних бібліотек, які надають деякі функціональні можливості Java SE, також існує безліч системних бібліотек у мовах C/C++, які допомагають створити основу для реалізації функціональності. Ці системні бібліотеки, насамперед, відповідають за функції, такі як малювання графіки, відтворення звуку та доступ до бази даних. API надає доступ до цих бібліотек через класи Java, які використовуються для розробки ігор.

Фреймворк Android забезпечує зв'язок між системними бібліотеками та середовищем виконання, створюючи прив'язку до платформи Android. Ця структура керує додатками і надає складне середовище для їх функціонування. Розробники створюють додатки для цієї структури, використовуючи набір API-інтерфейсів в мові Java, які охоплюють такі області, як розробка користувацького інтерфейсу, фонові служби, повідомлення, управління

ресурсами, доступ до периферійних пристроїв і т.д. Всі ключові додатки, які поставляються з операційною системою Android (наприклад, клієнт електронної пошти), написані з використанням цих API. Додатки, будь то інтерфейси або фонові служби, можуть взаємодіяти з іншими додатками. Це з'єднання дозволяє одному додатку використовувати компоненти інших додатків. Наприклад, програма, яка зробила фотографію, може викликати інший компонент додатка для обробки цього зображення. Перший додаток може повторно використовувати цей компонент, наприклад, використовуючи камеру, вбудовану в додаток або фотогалерею. Цей механізм розрішує велику частину завдань для розробника і дозволяє настроювати поведінку різних аспектів платформи Android. Для розробки власного додатка необхідно ознайомитися з інструментарієм розробки програм Android Software Development Kit (SDK). SDK надає розробникам необхідні інструменти і ресурси для створення, компіляції та тестування додатків для платформи Android [12].

2.2 Емулятор для розробки додатків Android SDK

Розробка програм на платформі Android вимагає використання Java API високого рівня, який дозволяє створювати програми для кінцевих користувачів Android. Будуть розглянуті характеристики емулятора Android, основні компоненти Android і пакети SDK. Крім того, наведено деякі кодові фрагменти. Також будуть приведені кілька фрагментів коду.

Android SDK, який включає Android Studio і плагін Android Development Tools (ADT), є набором інструментів для розробки додатків на платформі Android. Android Studio є інтегрованою середовищем розробки (IDE), яке надає можливості створення, налагодження та тестування додатків на Java. Хоча Android SDK може використовуватися без ADT, тобто за допомогою командного рядка, ADT пропонує зручніші інструменти. Емулятор Android

підтримує обидва підходи і дозволяє запускати, відлагоджувати і перевіряти додатки. Завдяки емулятору, більшість розробки додатків може бути виконана без фізичного пристрою. Емулятор повністю відтворює основні характеристики пристрою, хоча деякі функції, такі як USB-підключення, робота камери і відео, симуляція навушників, батареї та Bluetooth, не можуть бути повністю імітовані.

Емулятор Android базується на відкритій технології QEMU (Quick Emulator). QEMU розроблена компанією Bellard і забезпечує емуляцію на рівні процесора. При використанні емулятора Android він імітує процесор на основі архітектури ARM (Advanced RISC Machine). ARM - це 32-розрядна мікропроцесорна архітектура зі скороченим набором інструкцій RISC (Reduced Instruction Set Computer). Характеризується простотою конструкції та високою продуктивністю завдяки скороченому набору команд. Емулятор використовує симуляцію версії Linux, що використовується в Android.

Архітектура ARM широко використовується в мобільних пристроях і вбудованих електронних системах, де важлива енергоефективність. Завдяки помірному енергоспоживанню, процесори ARM забезпечують тривалу роботу пристрою без швидкого розрядження акумулятора. Це робить їх ідеальними для мобільних телефонів, планшетів, смарт-годинників та інших пристроїв, які потребують тривалого часу автономної роботи. Архітектура ARM також приваблива для вбудованих систем, де обмежені розміри та ресурси вимагають ефективного використання енергії.

Багато наявних впродажу мобільних пристроїв мають процесори з цієї архітектурою. Наприклад, Apple Newton на базі процесора ARM6. Ігрові автомати, Nintendo DS і Game Boy Advance працюють на архітектурі ARM версії 4, яка використовує близько 30 000 транзисторів. Класичний Pentium містить 3,2 мільйона транзисторів [6].

2.3 Компоненти для користувача інтерфейсу ANDROID

Фреймворк інтерфейсу Android можна порівняти з іншими повнофункціональними структурами інтерфейсу користувача, що використовуються на локальних комп'ютерах. Цей інтерфейс є більш сучасним та асинхронним.

Фреймворк Android UI належить до четвертого покоління, якщо порівнювати його з попередніми розробками. Перше покоління включало традиційне програмування інтерфейсу Microsoft Windows на основі C та MFC (Microsoft Foundation Classes, класи базових бібліотек Microsoft на основі C++) [5]. Другим поколінням став фреймворк Swing UI на основі Java, який виявився більш гнучким, ніж MFC. Користувацькі інтерфейси, такі як Android UI, JavaFX, Microsoft Silverlight і Mozilla XML (XUL), є частиною четвертого покоління фреймворків UI, які є декларативними і підтримують незалежну тематику. Програмування інтерфейсу Android використовує декларативні визначення інтерфейсу у файлах XML. XML-означення завантажуються в програму користувацького інтерфейсу, подібно до підходу, що використовується у Windows. Навіть меню програми можна завантажити з XML-файлу. Екрани або вікна Android відомі як "дії" і включають різні типи елементів, які користувач повинен виконати, щоб логічно продовжити процес. Основними елементами в інтерфейсі Android є "подання". Подання можуть бути об'єднані в групи, відомі як ViewGroups. Для внутрішньої організації цих елементів використовується концепція "полотна" і взаємодії користувача з системою..

У композитних уявленнях Android, що включають види та групи видів, використовується спеціальна логіка компонента користувацького інтерфейсу.

Управління життєвим циклом вікон подій (діяльностей) є одним із ключових аспектів платформи Android. Система використовує протоколи, які дозволяють Android керувати станом вікон подій під час взаємодії користувача. Це означає, що Android може керувати процесом приховування,

відновлення, зупинки та закриття вікон подій залежно від потреб користувача та поточного стану додатка.Основні компоненти Android.

Фреймворк для користувача інтерфейсу Android є одним з ключових компонентів платформи Android, і він базується на різних інших компонентах, включаючи наміри (Intents). Намір – це важлива концепція, яка дозволяє зв'язувати різні компоненти додатків, взаємодіяти з системою та виконувати різноманітні дії. Наміри можуть використовуватись для обміну інформацією між процесами та регістрами додатків. Вони можуть містити додаткові дані, такі як текстові рядки, URL-адреси, об'єкти та інші параметри, що передаються між компонентами.

У прикладі, що представлений на рисунку 2.1, використовуючи намір, спонукаємо Android відкрити вікно, підходяще для відображення вмісту веб-сайту. В залежності від того, який браузер встановлений на мобільному пристрої, Android буде вибрати найбільш підходяще для відображення сайту.

```
public static void invokeWebBrowser (Activity activity)
{
Intent intent = new Intent (Intent. ACTION_VIEW):
intent. setData (Uri. parse («http://www.google.com»));
 activity. startActivity (intent);
}
```

Рисунок 2.1 – Приклад Intent

Крім того, широко підтримуються ресурси Android, які знайомі з векторними та растровими зображеннями, і підтримуються принаймні деякі відомі елементи, такі як визначення вигляду на основі XML. Використання ресурсів в цих рамках, здійснюється по-новому, за допомогою чого робота ресурсу стає більш простою, зрозумілою і зручною. Нижче наведено приклад, в якому автоматично генерується ID ресурсу і визначається в файлах XML (рис. 2.2).

```
public final class R {
public static final class attr { }
```

```
public static final class drawable{
public static final int myanimation=0x7f020001;
public static final int numbersl9=0x7f02000e:
}
public static final class id {
public static final int textViewIdl=Ox7f080003;
}
public static final class layout {
public static final int frame animations layout=0x7f030001;
public static final int main=0x7f030002;
}
public static final class string
{
public static final int hello=0x7f070000;
}
}
```

Рисунок 2.2 – Генерація ID ресурсу

Ці класи в Android автоматично отримують ідентифікатори, які можуть бути згенеровані автоматично або вказані у XML-файлах для елементів або груп. Цей підхід значно спрощує локалізацію додатків, оскільки можна легко визначити різні версії елементів і ресурсів для різних мов або локалей.

Ще однією інноваційною концепцією в Android є постачальник контенту. Постачальник контенту є абстракцією джерела даних, яке може бути представлене як емітер або REST-сервіс. Цей підхід дозволяє розробникам легко отримувати доступ до даних і передавати їх між різними компонентами додатка. В основі цього механізму лежить абстракція бази даних SQLite, що робить постачальників контенту потужним інструментом для розробки додатків з роботою з даними.

XML грає важливу роль у визначенні користувацьких інтерфейсів для Android. Він використовується для опису компонентів і ресурсів, таких як макети екранів, стилі, рядки тексту та інші елементи, що складають інтерфейс додатка. Використання XML дозволяє зручно і структуровано визначати елементи і їх властивості, що спрощує розробку та налагодження інтерфейсів додатків.. Нижче показано, як оголосити меню в XML-файлі (рис. 2.3).

```
<menu xmlns: android=«http://schemes. android.com/apk/res/android»>
<!-В этой группе используется категория, заданная по умолчанию.-->
```

```
<group android: id=«@+id/menuGroup_Main»>
<item android: id=«@+id/menu_clear»
android: orderInCategory=«10»
android: title=«clear» />
<item android: id=«@+id/menu_show_browser»
android: orderInCategory=«5»
android: title=«show browser» />
</group>
</menu>
```

Рисунок 2.3 – Оголошення меню в XML-файлі

Діалогові вікна в Android є асинхронними, що означає, що вони працюють у фоновому режимі, і програма продовжує працювати після їхнього відображення. Цей підхід відрізняється від синхронних модальних діалогів, які зупиняють виконання програми до закриття вікна. Перехід до асинхронного підходу може бути складним для розробників, які звикли до синхронної взаємодії з діалоговими вікнами. Однак асинхронні діалоги дозволяють підтримувати продуктивність програми, оскільки вони не блокують роботу програми.

Android також підтримує анімацію двох типів: проміжна анімація (двостороння анімація) і кадрова анімація. Проміжна анімація використовується для переходу між основними кадрами. У цих кадрах зображення відображаються проміжними змінами, які відбуваються протягом певного періоду часу. З іншого боку, кадрова анімація складається з послідовності кадрів, які намальовані один за одним з однаковим інтервалом часу. Android використовує обидва види анімації, а також функції зворотного виклику анімації, інтерполятори та матриці перетворень. Крім того, Android може описувати ці види анімації у файлі XML-ресурсів, що дозволяє легко керувати анімацією.

Стандартні матричні перетворення, такі як масштабування, переміщення та поворот зображення, підтримуються вбудованою графічною бібліотекою Android. Крім того, графічна бібліотека містить об’єкт камери, який може керувати проекцією та глибиною, що дозволяє імітувати тривимірні ефекти в

двовимірному інтерфейсі.

Android отримав можливість використовувати тривимірну графіку завдяки впровадженню стандарту OpenGL ES 1.0. Цей протокол базується на мові програмування C, як і традиційний OpenGL. Для доступу до OpenGL ES використовується Java-зв’язування, оскільки Android SDK написаний на мові Java. Використання OpenGL ES є необхідним для Java ME (Micro Edition), як вже зазначено в специфікації Java QSR 239. Для роботи з OpenGL ES в Android використовується той самий тип зв’язування Java.

На Android використовуються нові підходи, які стосуються ідеї доступу до інформації на пальцях користувача через домашню сторінку. Один з цих підходів - це використання "живих каталогів". За допомогою живих каталогів, колекція елементів може бути опублікована в папці, розташованій на головній сторінці веб-сайту. Вміст цієї колекції автоматично оновлюється разом зі змінами в базі даних. Нові дані можуть бути надіслані на пристрій з знімного носія або з Інтернету. Це дозволяє користувачам отримувати актуальну інформацію без необхідності активного пошуку або оновлення додатків.

Друга ідея, пов'язана з домашньою сторінкою, – це віджет домашнього екрану. Віджети домашнього екрану використовуються для відображення інформації на домашній сторінці за допомогою користувацького інтерфейсу віджета. Цю інформацію можна змінювати через певні проміжки часу. Прикладом таких даних можуть бути кілька повідомлень електронної пошти, що зберігаються на пристрої.

Інтегрований пошук Android – це третя ідея, пов'язана з використанням домашньої сторінки. Цей тип пошуку дозволяє знаходити інформацію як на пристрої, так і в Інтернеті. В Android ця функція може бути обмежена лише одним пошуком і може давати команди за допомогою управління пошуком.

Крім того, Android підтримує розпізнавання жестів, тобто розуміє, як пальці людини взаємодіють з пристроєм. Послідовність рухів пальців на екрані можна записати та зберігати як жест. Згодом програми можуть використовувати ці жести для виконання певних завдань або навігації.

Наприклад, використання жесту «змах вліво» може призвести до перемикання на попередню сторінку, а використання жесту «щільний дотик» може викликати контекстне меню. Це дозволяє користувачам виконувати різні дії за допомогою простих жестів, що полегшує взаємодію з програмами [8].

Окрім інструментів Android SDK, існують й інші незалежні інновації, які роблять процес розробки цікавим та легким. Деякі з них – XML/VM, PhoneGap і Titanium. Titanium дозволяє використовувати технологію HTML у програмуванні для браузера Android на основі WebKit [11].

2.4 Пакети, які є частиною Android SDK

Для отримання розуміння платформи Android важливо дослідити структуру пакетів Java, яка входить до складу Android SDK. Оскільки Android SDK відрізняється від стандартного дистрибутиву, важливо знати, які пакети підтримуються і які ні. Нижче наведено короткий огляд ключових пакетів, що входять до складу Android SDK.

Android.app – реалізує модель додатку Android. Основні класи включають додаток, який описує початкову і кінцеву семантику, а також ряд класів, пов'язаних з явищами, елементами управління, діалогами, вікнами попереджень і повідомлень.

Android.Bluetooth – містить набір класів, які дозволяють взаємодіяти з технологією Bluetooth в Android. Основними класами є BluetoothAdapter, BluetoothDevice, BluetoothSocket, BluetoothServerSocket та BluetoothClass. Клас BluetoothAdapter дозволяє керувати встановленим Bluetooth-адаптером на комп’ютері. Цей клас дозволяє увімкнути, вимкнути або запустити процес виявлення. Клас пристроїв Bluetooth представляє віддалений пристрій Bluetooth, з яким можна встановити зв’язок. Завдяки двом роз’ємам між пристроями можна створювати зв’язок Bluetooth. Клас Bluetooth показує тип підключеного пристрою Bluetooth.

Android.content – реалізує концепції, пов'язані з постачальниками контенту. Контент-провайдер дозволяє узагальнити обмін та зберігання даних. Крім того, цей пакет реалізує основні принципи намірів (intents) та уніфікованих ідентифікаторів ресурсів (URI) в Android.

Android. content. pm. – надає класи для роботи, пов'язаної з використанням менеджера пакетів. Він містить інформацію про дозволи, встановлені пакунки, встановлених постачальників, сервіси та компоненти, такі як спільні ресурси, а також встановлені програми.

Android.gesture – цей пакет містить всі класи та інтерфейси, необхідні для роботи з користувацькими жестами. Основні класи, що входять до цього пакету, включають Gesture, GestureLibrary, GestureOverlayView, GestureStore, GestureStroke і GesturePoint. Клас Gesture представляє собою колекцію GestureStrokes і GesturePoints. Жести зберігаються в бібліотеці GestureLibrary. Бібліотеки жестів зберігаються у GestureStore. Назви жестів обрані таким чином, щоб система могла ідентифікувати їх як конкретні дії..

Android. graphics – містить класи Canvas, Camera, Color, Matrix, Movie, Paint, Path, Rasterizer, Shader, SweepGradient nTypeFace.

Android. hardware – дозволяє використовувати так звані природні класи, призначені для роботи з камерою. Клас camera – це звичайний пристрій – камера, а клас android.graphics.Camera – це графічний концепт, який не має нічого спільного з реальною фізичною камерою.

Android.location – містить класи Address, GeoCoder, Location, LocationManager та LocationProvider. Клас Address є спрощенням мови XAL (розширювана мова адрес). Geocoder дозволяє дізнатися адресу за координатами об'єкта (широта і довгота), і навпаки. Location надає інформацію про широту та довготу.

Android.media – містить класи MediaPlayer, MediaRecorder, Ringtone, AudioManager та FaceDetector. Клас MediaPlayer використовується для відтворення аудіо та відео в режимі потокової передачі. Ringtone відповідає за відтворення коротких звукових сигналів, які можна використовувати як

рингтони або сповіщення. AudioManager дозволяє керувати аудіо-параметрами пристрою. FaceDetector використовується для виявлення обличчя на зображеннях.. AudioManager відповідає за регулювання гучності. FaceDetector можна використовувати для розпізнавання людських облич на точкових (растрових) зображеннях [9].

Перераховані вище пакети є дуже важливими при роботі з Android. Цей список дає уявлення про глибоку структуру платформи Android.

Загалом, Android Java API включає в себе більше 40 пакетів і більше 700 класів. Проте, всі ці різноманітні пакети створюють потужну обчислювальну платформу, спеціально розроблену для розробки програм для мобільних пристроїв [10].

2.5 Use case діаграма

На рисунку 2.4 представлена Use case діаграма.

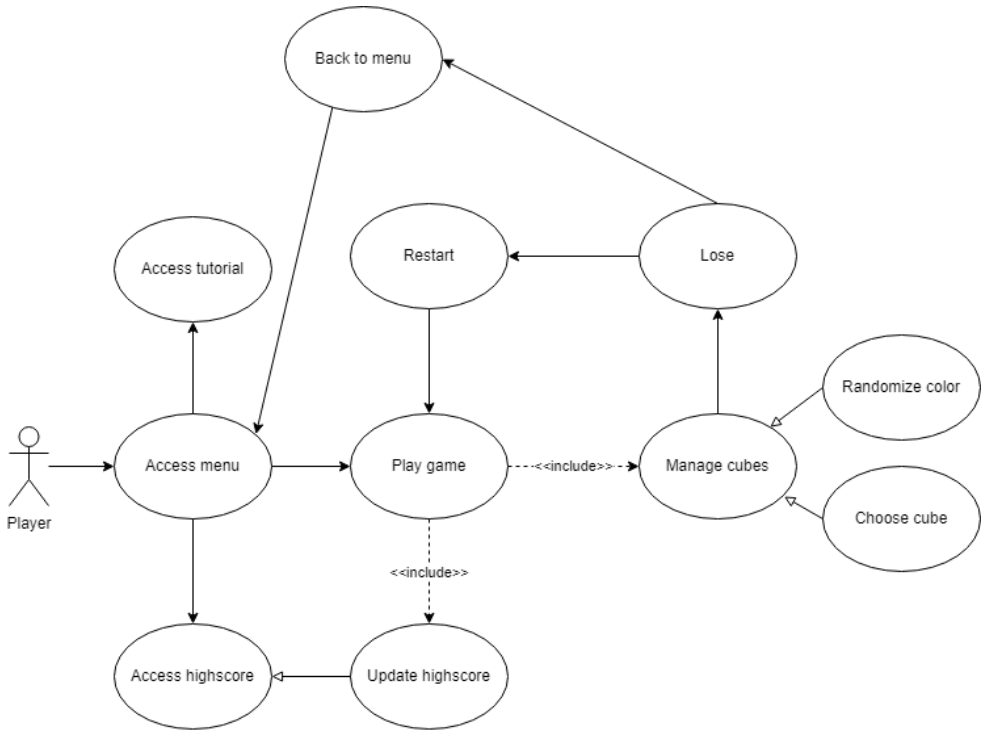

Рисунок 2.4 – Use case діаграма

Актори: Гравець.

Основний потік подій:

а) гравець запускає гру;

б) система завантажує головну сцену («main scene»);

в) на головній сцені відображаються об'єкти: Main Camera, Buttons, Canvas, EventSystem, Cube, Directional Light, Click Audio, Background;

г) користувач взаємодіє з кнопками на екрані головного меню:

1) користувач натискає на кнопку «Play»;
   - система переходить на сцену гри («play scene»);
   - система починає гру;
2) користувач натискає на кнопку «How To»;
3) система відкриває меню з поясненнями про правила гри;

д) на сцені гри відображаються об'єкти: Main Camera, Main Cube, Canvas, EventSystem, Directional Light (1, 2, 3), Directional Light, прихований об'єкт «Panel Lost» і Background;

е) під час гри користувач взаємодіє з об'єктом Main Cube, який містить скрипт гри;

ж) у разі програшу:

1) система викликає прихований об'єкт «Panel Lost»;
2) об'єкт «Panel Lost» активується і відображає вікно з опцією перезапуску гри (кнопка «Restart»), повернення в головне меню (кнопка «Home») і кнопку Instagram для переходу на сторінку автора гри;

з) після кожної гри гравець досягає певного результату;

1) система порівнює поточний результат із рекордом (Highscore), збереженим у системі;
2) якщо поточний результат перевищує збережений рекорд, виконується наступний крок, інакше гра триває без змін;
3) система оновлює значення рекорду (Highscore) новим

досягнутим результатом;

4) система зберігає оновлене значення рекорду в сховищі даних, щоб воно було доступне в наступних сесіях гри.

Альтернативні дії:

5) якщо поточний результат не перевищує збережений рекорд (Highscore), то система не оновлює значення рекорду.

Ключові об'єкти, що взаємодіють:

– гравець;
– головне меню;
– сцена гри;
– кнопка «Play»;
– кнопка «How To»;

Об'єкти на сцені гри: Main Camera, Main Cube, Canvas, EventSystem, Directional Light (1, 2, 3), Directional Light, прихований об'єкт «Panel Lost» і Background.

Цей опис представляє основний потік подій і ключові об'єкти, що взаємодіють під час гри [4].

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробку ПЗ почнемо з графічного інтерфейсу. Це пов’язано з особливостями програмування для Android. Для того, щоб взаємодіяти з елементами вікна, програмний модуль підключається до необхідних елементів за допомогою ідентифікаторів. При цьому елементи інтерфейсу в головному вікні повинні бути створені в першу чергу.

3.1 Графічний інтерфейс користувача

Екран головного меню являє собою фон з кнопкою “Play” для переходу до екрану самої гри та кнопкою “How To” для пояснювання правил гри, а також лічильник максимального рахунку (рис. 3.1).

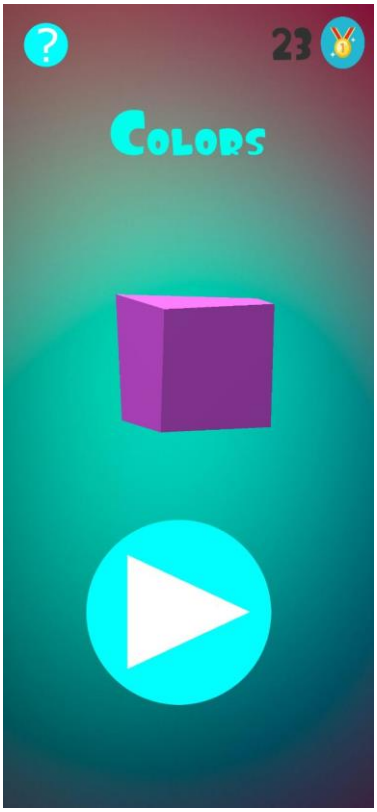

Рисунок 3.1 – Main Menu

При натисканні на кнопку “Play” відбувається перехід до сцени play та початок гри (рис. 3.2).

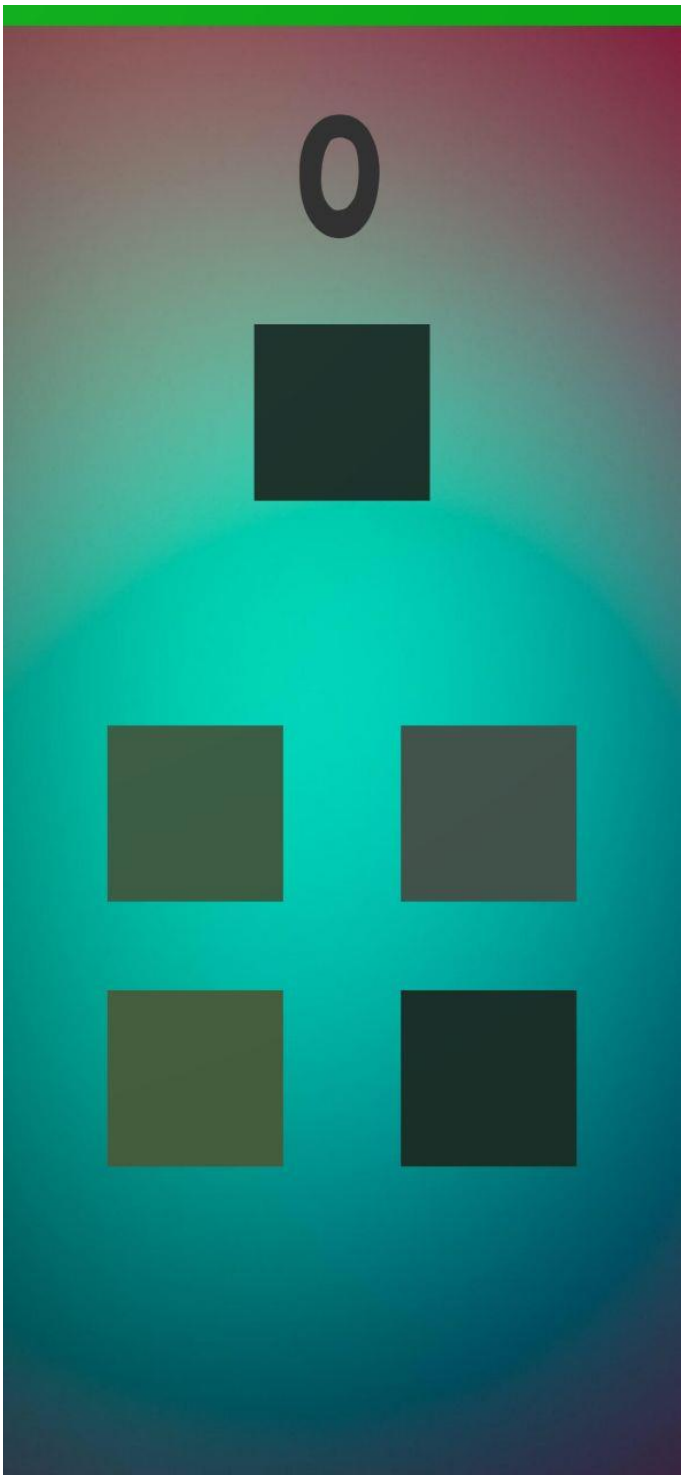

Рисунок 3.2 – Сцена Play

При натисканні на кнопку “How To” відкривається меню з поясненням як грати(рис. 3.3).

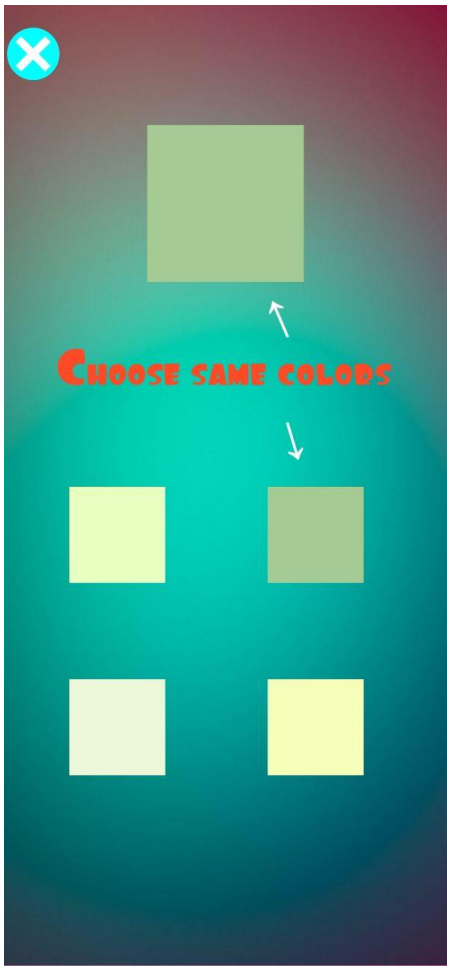

Рисунок 3.3 – How To

3.2 Робота програми

При запуску гри починає роботу сцена main. У сцені main реалізовані такі об’єкти як: Main Camera, Buttons, Canvas, EventSystem, Cube, Directional Light, Click Audio, Background (рис. 3.4).

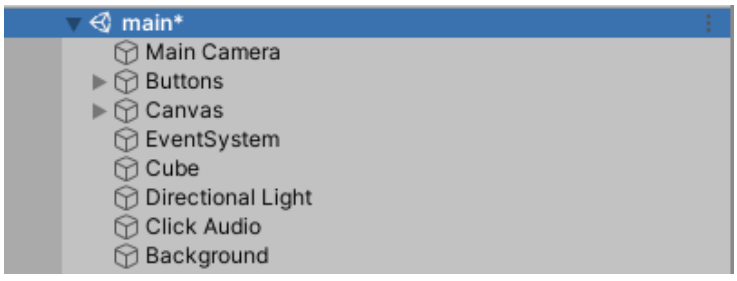

Рисунок 3.4 – Об’єкти сцени Main

В об’єкті Buttons лежать ще кілька об’єктів які відповідають за кнопки в запропонованій грі (рис. 3.5).

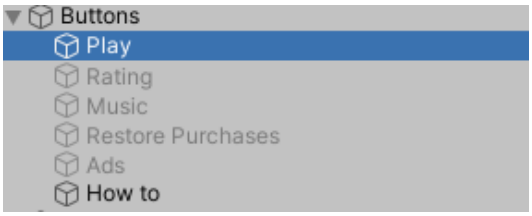

Рисунок 3.5 – Об’єкти в Buttons

З кнопкою Play пов’язаний скріпт переходу на сцену play та початок гри та зміну зображення кнопки при натисканні (рис. 3.6).

```
public class Play : MonoBehaviour
{
   public Sprite Play_or_Next1, Play_or_Next;

   void OnMouseDown()
   {
      GetComponent<SpriteRenderer>().sprite = Play_or_Next1;

   }
   void OnMouseUp()
   {
      GetComponent<SpriteRenderer>().sprite = Play_or_Next;
   }
   void OnMouseUpAsButton()
   {
      if (PlayerPrefs.GetString(«Music») != »no»)
         GameObject.Find(«Click Audio»).GetComponent<AudioSource>().Play();
      switch (gameObject.name)
      {
         case »Play»:
            Application.LoadLevel(«play»);
            break;
      }
   }
}
```

Рисунок 3.6 – Скріпт переходу на сцену play

В сцені play лежать такі об’єкти як: Main Camera, Main Cube, Canvas, EventSystem, Directional Light (1,2,3), Directional Light, скритийоб’єкт Panel Lost та Background (рис. 3.7).

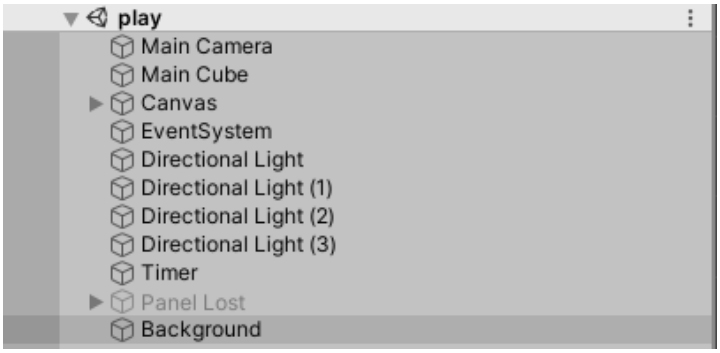

Рисунок 3.7 – Об’єкти сцени play

В об’єкті Main Cube лежить скрипт самої гри, лістінг представлено в Додатку А.

Після програшу викликається скритий об’єкт Panel Lost, який активується та показує вікно програшу з кнопками Restart, Home та кнопкою Instagram для переходу на Інстаграм автора гри (рис. 3.8).

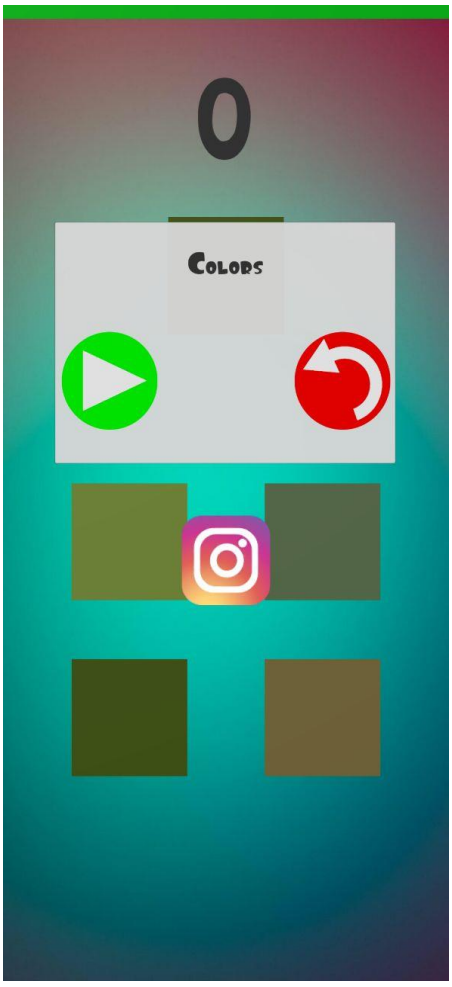

Рисунок 3.8 – Перехід на Інстаграм

ВИСНОВКИ

Результатом виконання дипломного проєкту є Android – додаток «Colors».

Під час роботи було виконано наступні завдання:

– проаналізовано організацію електронного документообігу;
– описано особливості інформаційних систем та систем електронного документообігу;
– розроблено проєкт Android-додатку;
– розроблено програмну реалізацію Android-додатку.

Було спроєктовано та створено Android-додаток “Colors”. За допомогою якого можна цікаво та з користю провести свій вільний час.

Переваги додатку: простий інтерфейс; безоплатне завантаження; можливість розвивати свою реакцію.

Android-додаток може бути адаптований під будь який екран сучасного смартфона.

ПЕРЕЛІК ПОСИЛАНЬ

1. Опануй програмування граючись. Вивчаємо Android. Перший курс. URL: http://developer.alexanderklimov.ru/android/ (дата звернення: 02.04.2023).

2. Skillbox. Як створити 2D – гру на Unity. URL: https://skillbox.ru/media/code/kak_sozdat_prostuyu_2d_igru_na_unity (дата звернення: 03.04.2023).

3. Habr. Приклад створення простої 2D гри для Android з використанням ігрового рушія Unity. URL: https://habr.com/ru/post/283186/ (дата звернення: 03.04.2023).

4. Unity.Unity для мобільних пристроїв. URL: https://unity.com/ru/solutions/ mobile (дата звернення: 04.04.2023).

5. Microsoft. Visual Studio 2019. URL:https://visualstudio.microsoft.com (дата звернення: 05.04.2023).

6. Developers. Android Studio. URL: https://developer.android.com/studio (дата звернення: 06.03.2023).

7. DiMarzio J. F. Android Game Recipes: A Problem-Solution Approach. Apress, 2013. 256 p.

8. Habr. Досвід створення гри для Android поодинці з нуля і як її зафічерили на Google Play. URL: https://habr.com/ru/post/439266/ (дата звернення: 07.03.2023).

9. Wikipedia. Android. URL: https://ru.wikipedia.org/wiki/Android (дата звернення: 07.03.2023).

10. Android. Офіційний сайт Android. URL: https://www.android.com/intl/ru_ru/ (дата звернення: 07.03.2023).

11. Довідка Google. Як керувати дозволами для додатків в ОС Android 6.0 пізніших версій. URL: https://support.google.com/googleplay/answer/6270602?hl=ru (дата звернення:

07.04.2023).

12. Androidinsider. Що таке Android? URL: https://androidinsider.ru/smartfony/chto-takoe-android.html (дата звернення: 07.04.2023).

ДОДАТОК А

Приклад коду скриптів

А.1 Лістінг скрипта MainCube

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;
using GoogleMobileAds.Api;

public class GameCntrl : MonoBehaviour



{
        public GameObject plost;

        public GameObject colBlock;
        public Vector3[] positions;
        private GameObject block;
        private GameObject[] blocks = new GameObject[4];

        private int rand, count;
        private float rCol, gCol, bCol;
        public Text score;
        private static Color aColor;

        [HideInInspector]
        public bool next, lose;
```

```
	private const string banner = «ca-app-pub-
5206105264433301/1156606441»;
	private const string gameOverAD = «ca-app-pub-
5206105264433301/1051776615»;

	void Start()
	{
		count = 0;
		next = false;
		lose = false;
		rand = Random.Range(0, positions.Length);
		for (int i = 0; i < positions.Length; i++)
		{
			blocks[i] = Instantiate(colBlock, positions[i],
Quaternion.identity) as GameObject;
			if (rand == i)
				block = blocks[i];
		}
		block.GetComponent<RandCol>().right = true;
		BannerView bannerV = new BannerView(banner,
AdSize.Banner, AdPosition.Bottom);
		AdRequest request = new
AdRequest.Builder().AddTestDevice(AdRequest.TestDeviceSimulator).AddTest
Device(«63DE30FF6B7C472B»).Build();
		bannerV.LoadAd(request);
	}

	void Update()
	{
```

```
            if (lose)
                playerLose();
            if (next && !lose)
                nextColors();
            if (PlayerPrefs.GetInt(«Score») < count)
                PlayerPrefs.SetInt(«Score», count);
        }

        void nextColors()
        {
            if (PlayerPrefs.GetString(«Music») != «no»)
                GetComponent<AudioSource>().Play();
            count++;
            score.text = count.ToString();
            aColor = new Vector4(Random.Range(0.1f, 1f),
Random.Range(0.1f, 1f), Random.Range(0.1f, 1f), 1);
            GetComponent<Renderer>().material.color = aColor;
            next = false;

            if (count < 5)
            {
                rCol = 0.5f;
                gCol = 0.5f;
                bCol = 0.5f;
            }
            else if (count >= 5 && count < 10)
            {
                rCol = 0.3f;
                gCol = 0.3f;
                bCol = 0.2f;
```

```
                }
                else if (count >= 10)
                {
                        rCol = 0.2f;
                        gCol = 0.2f;
                        bCol = 0.17f;
                }

                // New colors for blocks
                rand = Random.Range(0, positions.Length);
                for (int i = 0; i < positions.Length; i++)
                {
                        if (i == rand)

        blocks[i].GetComponent<Renderer>().material.color = aColor;
                        else
                        {
                                float r = aColor.r + Random.Range(-rCol, rCol) >
1f ? 1f : aColor.r + Random.Range(-rCol, rCol);
                                float g = aColor.g + Random.Range(-gCol, gCol)
> 1f ? 1f : aColor.g + Random.Range(-gCol, gCol);
                                float b = aColor.b + Random.Range(-bCol, bCol)
> 1f ? 1f : aColor.b + Random.Range(-bCol, bCol);

        blocks[i].GetComponent<Renderer>().material.color = new Vector4(r, g,
b, aColor.a);
                        }
                }
        }
```

```
        void playerLose()
        {
            InterstitialAd ad = new InterstitialAd(gameOverAD);
            AdRequest request = new
AdRequest.Builder().AddTestDevice(AdRequest.TestDeviceSimulator).AddTest
Device(«63DE30FF6B7C472B»).Build();
            ad.LoadAd(request);
            if (ad.IsLoaded()) ad.Show();
            print(«Player lose»);
            plost.SetActive(true);
            if (PlayerPrefs.GetString(«Music») == «no»)
                plost.GetComponent<AudioSource>().mute = true;
        }
    }
```