ЗВІТ З ПЕРЕВІРКИ НА ПЛАГІАТ

ЦЕЙ ЗВІТ ЗАСВІДЧУЄ, ЩО ПРИКРПЛЕНА РОБОТА

Мірошникова ЯО ІПЗ-119

БУЛА ПЕРЕВІРЕНА СЕРВІСОМ ДЛЯ ЗАПОБІГАННЯ ПЛАГІАТУ

MY.PLAG.COM.UA І МАЄ:

СХОЖІСТЬ

4%

РИЗИК ПЛАГІАТУ

57%

ПЕРЕФРАЗУВАННЯ

0%

НЕПРАВИЛЬНІ ЦИТУВАННЯ

0%

Назва файлу: Мірошникова ЯО ІПЗ-119.docx

Файл перевірено: 2023-06-16

Звіт створено: 2023-06-16

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ПРИВАТНЕ АКЦІОНЕРНЕ ТОВАРИСТВО «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ» (WWW.ZIEIT.EDU.UA)

Кафедра інформаційних технологій

ДО ЗАХИСТУ ДОПУЩЕНА

Зав. кафедрою _____

д.е.н., доц. Левицький С.І.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

РОЗРОБКА ІГРОВОГО МОБІЛЬНОГО ЕКОДОДАТКУ З АНТРОПОМОРФНИМИ ПЕРСОНАЖАМИ

Виконав
ст. гр. ІПЗ-119 _____ Я.О. Мірошникова

Керівник
професор _____ С.О. Сабанов

Запоріжжя
2023

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ

студенту гр. ІПЗ-119, спеціальності 121 - «Інженерія програмного забезпечення»

Мірошниковій Ярославі Олександрівні

1.Тема: «Розробка ігрового мобільного екододатку з антропоморфними персонажами»

затверджена наказом № 02-10 від 27 січня 2023 р.

2. Термін здачі студентом закінченої роботи: 20 червня 2023 р.

3. Перелік питань, що підлягають розробці:

1. (library.econom.zp.ua) Провести огляд предметної області, ознайомитися з літературою та інтернет-джерелами, що присвячені тематиці роботи.

2. Розглянути типові реалізації антропоморфізму в сучасних ігрових додатках, проаналізувати особливості використання в них програмних та апаратних ресурсів.

3. Ознайомитися із засобами розробки з метою вибору оптимального інструментарію створення проекту.

4. Здійснити проектування програмного додатку з урахуванням функціоналу.

5. Розробити та реалізувати алгоритм функціонування системи.

6. Відлагодити програмний комплекс.

7. Оформити звіт за результатами роботи (library.econom.zp.ua)

К А Л Е Н Д А Р Н И Й Г Р А Ф І К
підготовки бакалаврської дипломної роботи
здобувачем освіти інституту ЗІЕІТ денної форми навчання
гр. ІПЗ-119 Мірошниковою Ярославою Олександрівною
2022-2023 навчальний рік

№ етапу	Зміст	Терміни виконання (zieit.dp.ua)	Готовність по графіку %, підпис керівника	Підпис керівника про повну готовність етапу, дата
1	Збір практичного матеріалу за темою дипломної роботи	16.01.23-11.02.23		
2	І атестація І розділ бакалаврської дипломної роботи	27.03.23-31.03.23		
3 (library.econom.zp.ua)	ІІ атестація ІІ розділ бакалаврської дипломної роботи	24.04.23-28.04.23		
4	ІІІ атестація ІІІ розділ бакалаврської дипломної роботи, висновки та рекомендації, додатки, реферат	22.05.23-26.05.23		
5	Перевірка дипломної роботи на оригінальність	15.05.23-12.06.23		
6	Доопрацювання бакалаврської дипломної роботи, підготовка презентації, отримання відгуку керівника і рецензії	29.05.23-12.06.23		
7	Попередній захист бакалаврської дипломної роботи	12.06.23-18.06.23		
8 (77.93.36.128)	Подача бакалаврської дипломної роботи на кафедру	за 3 дні до захисту		
9	Захист бакалаврської дипломної роботи (zieit.dp.ua)	19.06.23-24.06.23		

Керівник _____ С.О. Сабанов

Здобувач освіти _____ Я.О. Мірошникова

РЕФЕРАТ

Бакалаврська дипломна робота містить: 53 сторінок, 18 рисунків, 17 першоджерел та 1 додаток.

Об’єктом розробки є мобільний додаток для платформи Android.

Мета роботи: проєктування та розробка мобільного додатку на екологічну тематику з використанням антропорфних персонажів.

У роботі детально розглядаються всі етапи розробки проекту, починаючи від проєктування та аналізу вимог до розробки додатку з урахуванням необхідного функціоналу, тестування та налагодження.

В роботі використовується середа розробки Unity, яка підтримує мову програмування C#.

Окрема увага приділена необхідності розробляти додатки на екологічну тематику, розробці функціоналу та скриптів для роботи з ним.

Результати, отримані в даній роботі, можуть бути використані для подальшої розробки мобільних додатків та вдосконалення даного додатку.

UNITY, C#, ANDROID, АНТРОПОРФІЗМ, ЕКОДОДАТОК

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

Добавлено примечание ([21]): Відсутнє в змісті

Слово / словосполучення	Скорочення
A	
Application Programming Interface	API (77.93.36.128)
Android Package Kit	APK
C	
Cascading Style Sheets	CSS
H	
HyperText Markup Language (www.auhd.site)	HTML
I	
Integrated Development (77.93.36.128) Environment)	IDE
S	
Software Development Kit	SDK
VS	
Visual Studio	VS
eXtensible Markup Language	XML (www.auhd.site)
eXtensible Stylesheet Language Transformations	XSLT
Операційна система	ОС
П	
Персональний комп’ютер	ПК

ВСТУП

Екологічна тематика в мобільних додатках – це актуальна проблема, яка потребує уваги та досліджень. Мобільні додатки стали неодмінною частиною нашого життя, і їх використання розширюється з кожним днем. Тому, включення екологічної тематики у мобільні додатки може бути корисним інструментом для залучення уваги користувачів до проблеми екології та розширення свідомості про неї.

У наш час, на ринку мобільних додатків можна знайти багато застосувань, які мають екологічну тематику. Такі додатки можуть надавати користувачам інформацію про екологічні проблеми та навчати їх, як зменшити свій вплив на навколишнє середовище. Також, додатки можуть допомагати користувачам знаходити екологічно чисті місця для відпочинку, а також допомагати у сортуванні сміття.

Загальне поняття антропоморфізму, яке використовується у мобільних додатках з екологічною тематикою, є важливим елементом для залучення користувачів та підвищення їх зацікавленості до проблем екології. Антропоморфізм дає можливість створити персонажів, які будуть сприйматися користувачами як рівноправні істоти зі своїми власними потребами та проблемами. Такі персонажі можуть бути створені на основі тварин та птахів, що мають проблеми з виживанням через людську діяльність.

Екологічна тематика в мобільних додатках є досить актуальною та важливою тому розробка таких додатків сприяє підвищенню екологічної свідомості населення та формуванню здорового способу життя. Додатки на екологічну тематику можуть бути розроблені для різних платформ та мати різну спрямованість, а саме: навчальну, інформаційну, моніторингову, сприяння екологічним ініціативам, тощо. Що стосується антропоморфізму в мобільних додатках, то він може бути використаний як інструмент для підвищення емоційної привабливості додатків та привертання уваги

користувачів. Однак, варто враховувати етичні аспекти та не використовувати антропоморфізм для відволікання користувачів від екологічних проблем.

Антропоморфізм може бути корисним для створення більш цікавого та персоналізованого досвіду для користувачів мобільних додатків. Реалізація віртуальних персонажів може зробити взаємодію з додатком більш емоційно насиченою та забезпечити додаткову мотивацію для виконання завдань. Проте, при реалізації антропоморфізму в мобільних додатках, важливо враховувати потенційні недоліки та ризики, які можуть призвести до погіршення взаємодії з додатком та негативного впливу на користувача. Наприклад, недостатня точність або некоректна поведінка віртуальних персонажів можуть призвести до незадоволення користувача та погіршення його досвіду використання додатка. Узагалі, реалізація антропоморфізму в мобільних додатках може бути корисним інструментом для покращення взаємодії з користувачами та забезпечення більш персоналізованого та емоційно насиченого досвіду. Проте, важливо забезпечувати високу якість виконання додатком своїх функцій та уникати потенційних недоліків та ризиків, які можуть негативно впливати на користувача.

Наразі екологічна тематика в мобільних додатках є досить актуальною, тому що екологічні проблеми стають все більш серйозними і потребують термінового вирішення. За допомогою мобільних додатків можна навчити користувачів зберігати ресурси та робити екологічні вибори. Екологічні мобільні додатки можуть сприяти збільшенню свідомості людей про екологічні проблеми, а також допомогти їм в прийнятті правильних рішень.

У випадку екологічних мобільних додатків слід враховувати багато різних факторів, таких як (openarchive.nure.ua) дизайн та зручність використання, функціональні можливості, ефективність та зручність навігації, збереження даних тощо. Для досягнення найбільш ефективних результатів варто обрати оптимальну комбінацію цих факторів.

РОЗДІЛ 1
ЕКОЛОГІЧНА ТЕМАТИКА ТА РЕАЛІЗАЦІЯ АНТРОПОМОРФІЗМУ В МОБІЛЬНИХ ДОДАТКАХ

1.1. Екологічна тематика в мобільних додатках

В останні роки наше суспільство все більше звертає увагу на екологічні проблеми та катастрофи антропогенного характеру, тому екологічна тематика все частіше стає провідною в розробці різноманітних за функціональною направленістю додатків. Мобільні додатки можуть бути ефективним інструментом для поширення інформації про екологічні проблеми та розширення аудиторії, яка звертає на них увагу.

У мобільних додатках екологічна тематика може мати декілька форматів: від навчальних додатків, що розповідають про екологію, до додатків-екопомічників, що допомагають зменшити екологічний слід користувача на природу. Додатки, які спрямовані на зменшення екологічного впливу, можуть допомогти людям бути більш свідомими та дбайливими до довкілля. Наприклад, додатки-календарі можуть нагадувати користувачам про дні, коли треба виводити сміття або проводити енергозберігаючі заходи. Додатки-геолокації можуть допомогти користувачам знайти найближчу зупинку громадського транспорту або станцію велопрокату. Додатки для екологічного пішохідного туризму можуть допомогти людям дізнатися більше про різноманітні екосистеми, які потрібно захищати та зберігати.

Зокрема, можна виділити такі програми, як Trash-It, Garbage 31, Сортуй, Litterati, Charitymiles, Ecoinspector UA, go Recycle, Save Eco Bot, Love Food Hate Waste, Stockfactory, Dirty Dozen та багато інших. Аналіз показав, що практично всі подібні додатки орієнтовані на людей які вже зацікавлені в збереженні навколишнього середовища, одночасно з цим додатки, які

мотивують людей, заохочуючи дізнатись більше про нинішню екологічну ситуацію в світі, практично не зустрічаються.

Крім мобільних додатків з екологічною тематикою, існують також різні інтерактивні проекти, що дозволяють користувачам бути більш активними учасниками вирішення екологічних проблем. Наприклад, є проекти, які дозволяють користувачам допомагати у зборі відходів у природі, ліквідації місць незаконного сміттєзвалища, або ж залучатися до організації проектів з підтримки екологічних ініціатив.

Однією з переваг мобільних додатків є те, що вони можуть допомогти взаємодіяти з місцевими урядовими органами та екологічними організаціями, що дозволить користувачам бути більш активними учасниками вирішення екологічних проблем в їхньому регіоні. Мобільні додатки також можуть допомогти у співпраці між урядовими органами та громадськістю, допомагаючи збирати дані та інформацію про екологічні проблеми, що дозволить урядовим органам більш ефективно вирішувати ці проблеми. Важливо відзначити, що мобільні додатки з екологічною тематикою можуть бути ефективними інструментами не лише для користувачів, але й для бізнесу. Компанії можуть використовувати ці додатки для просування своїх екологічних продуктів та послуг, або ж для збору даних та отримання зворотного зв'язку від користувачів щодо їхнього відношення до екології та продуктів, які вони надають.

Нові мобільні додатки, які сприяють екологічному способу життя, з'являються щодня. Загалом, розвиток мобільних додатків, спрямованих на покращення екологічної ситуації, може забезпечити значний внесок у боротьбу з екологічними проблемами. Однак, варто зазначити, що технології не є панацеєю і самі по собі не здатні розв'язати всі екологічні проблеми.

Отже, можна зробити висновок, що мобільні додатки можуть допомогти в боротьбі з екологічними проблемами, проте не можна розглядати їх як єдине рішення проблеми. І водночас, варто підтримувати та використовувати такі

додатки, які допомагають зберегти навколишнє середовище та навіть покращити його стан.

1.2. Загальне поняття антропоморфізму

З іншого боку, в ході дослідження увагу автора привернула наявність великої кількості ігрових проектів з антропоморфними персонажами. Антропоморфізм – наділення предметів, тварин, явищ природи, міфічних істот та небесних тіл зовнішністю, пропорціями і фізичними властивостями людини [1].

Ми також можемо розглядати антропоморфізм як спосіб відображення екологічних проблем. Це дає змогу використовувати тваринний антропоморфізм для передачі інформації про екологічні проблеми, що стосуються тваринного світу. Антропоморфізм також може бути використаний для створення ідентичності користувача з персонажами, що може допомогти залучити більше людей до розв'язання екологічних проблем.

Загальне поняття антропоморфізму, яке використовується у мобільних додатках з екологічною тематикою, є важливим елементом для залучення користувачів та підвищення їх зацікавленості до проблем екології. Антропоморфізм дає можливість створити персонажів, які будуть сприйматися користувачами як рівноправні істоти зі своїми власними потребами та проблемами. Такі персонажі можуть бути створені на основі тварин та птахів, що мають проблеми з виживанням через людську діяльність.

Антропоморфізм в мобільних додатках може мати різні форми, від простих елементів, які надають імена і персональність, до складних алгоритмів, які враховують поведінку користувача для визначення оптимальної стратегії взаємодії з ним. Основна мета антропоморфізму в мобільних додатках - зробити користування додатком більш приємним та

ефективним. Одним з прикладів антропоморфізму в мобільних додатках є голосовий помічник, такий як Siri чи Google Assistant. Ці помічники здатні відповідати на запитання користувача та виконувати дії, такі як надсилання повідомлень або запуск додатків. Більш того, вони володіють індивідуальними рисами характеру, такими як гумор, що дозволяє їм взаємодіяти з користувачами на більш особистому рівні. Інший приклад антропоморфізму в мобільних додатках - це використання символів, таких як емісії, що відображають різні емоції та стани настрою. Емодзі дозволяють користувачам виразити свої емоції та почуття в текстових повідомленнях, що додає більше емоційної виразності в комунікації. Крім того, деякі мобільні додатки використовують антропоморфізм для створення віртуальних персонажів, які допомагають користувачам у виконанні різних завдань. Наприклад, додаток Duolingo використовує віртуального пташеня, яке надає користувачам зворотний зв'язок та мотивацію для вивчення мови. Віртуальні персонажі можуть мати свої унікальні риси характеру, що дозволяє їм створювати більш персоналізований досвід для користувачів.

Реалізація антропоморфізму в мобільних додатках може мати позитивний вплив на користувача. Використання символів та віртуальних персонажів може зробити взаємодію з додатком більш цікавою та забезпечити мотивацію для виконання завдань. Голосові помічники можуть зробити користування додатком більш зручним та ефективним. Однак, недоліки антропоморфізму також можуть бути присутні. Якщо віртуальний персонаж або голосовий помічник не відповідає на запитання користувача або надає неправильну інформацію, це може призвести до погіршення взаємодії з додатком та негативного впливу на користувача.

Антропоморфізм вперше зустрічається в молитвах та релігійних текстах, боги стародавніх єгиптян були напівлюдинами і напівтваринами. В сьогоденні це трансформувалось у спільноту під назвою фурі. Фурі – люди, які активно цікавляться антропоморфними персонажами. Часто цих персонажів

створюють самі учасники спільноти [2].

1.3. Фурі-фендом та фурі-івенти в сучасному світі

Фурі-фендом – це підкультура, що пов'язана з антропоморфізмом, в якій учасники приймають тваринний образ і відтворюють його через костюми, музику та інші види мистецтва. Фурі-івенти – це зустрічі учасників фурі-фендому, де вони можуть демонструвати свої костюми, взаємодіяти один з одним та спілкуватися про різноманітні теми. Фурі-фендом та фурі-івенти стають все більш популярними і збирають все більше прихильників.

Фурі-фендом має пряме відношення до екологічних проблем в контексті тварин. Наприклад, на фурі-конвентах можуть бути представлені інформаційні стенди або презентації про екологічні проблеми, що стосуються фурі-спільноти. Такі заходи можуть бути корисні для популяризації знань про екологію та залучення уваги до проблем охорони навколишнього середовища. Крім того, можуть проводитись конкурси та ігри, пов'язані з екологічною тематикою, які допомагають підвищити свідомість учасників про проблеми довкілля та шляхи їх вирішення.

Фурі також унікальні тим, що всі вони підходять до фендому з різних точок зору. У той час як одні люблять одягатися у своїх фурсонів (антропоморфний персонаж, вигаданий конкретною людиною для уособлення своїх рис), інші віддають перевагу створенню або споживанню фан-арту, натхненного тваринами [3].

Треба зауважити, що антропоморфізм досить часто зустрічається в іграх, де його використовують для створення цілих рас. Антропоморфних персонажів використовують, бо по відношенню до них гравці легше відчувають емпатію, асоціюють себе з цими персонажами [4]. Оскільки в додатку слід зробити акцент на почуттях користувачів до тварин, було вирішено зробити усіх персонажів антропоморфними – для полегшення емоційного відгуку на екологічну тематику розробки.

1.4. Реалізація антропоморфізму в іграх

В останні роки індустрія ігор зазнала змін, ми все частіше бачимо антропоморфних персонажів у іграх. Раніше це проявлялось в наданні персонажам схожої зовнішності але деякі риси тварин залишались присутніми в дизайні. Наприклад, антропоморфні персонажі мали анатомію людей, але також мали хвости, вуха, лапи, кігті, носи тварин. Поступово тваринні риси почали відходити на другий план при розробці дизайну персонажів (рисунок 1.1).

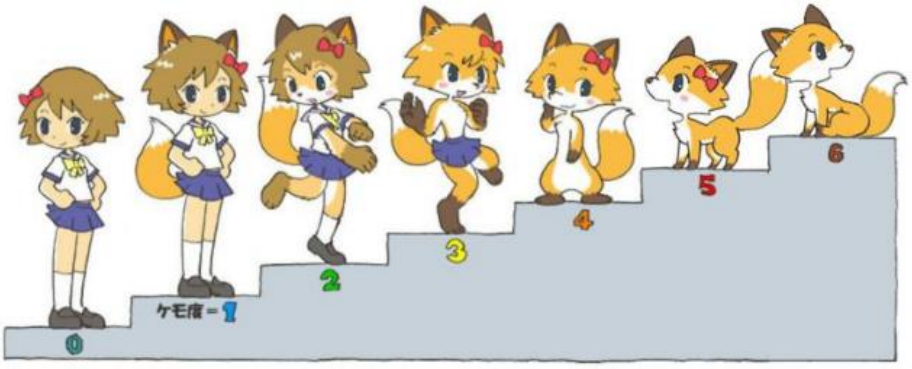

Рисунок 1.1 – Варіанти дизайну персонажів з різною кількістю антропоморфних і тваринних рис

Хоча додатків-аналогів у яких використовуються антропоморфні персонажі можна назвати дуже багато, поєднання антропоморфізму і екологічної тематики можна побачити вкрай рідко. На думку автора, це є великим упущенням, адже використовувати антропоморфні версії тварин та екологічних проблем аби донести до суспільства важливість збереження світу – один із найвдаліших варіантів.

1.5. Огляд аналогів

Для розробки гри необхідно проаналізувати якомога більше аналогів існуючих ігор.

«Night in the Woods». Фабулу гри можна викласти так: Мей Боровскі, головна героїня гри, закінчує коледж, повертається додому в рідне містечко шахтарів Поссум-Спрінгс. Вона намагається поліпшити своє життя та відновити зв'язки зі старими друзями, яких вона покинула. Але після довгої відсутності дім здається іншим, а її друзі виросли, змінились та мають інші інтереси. Вона блікає наодинці лісом і помічає що там щось є [5].

Рисунок 1.2. – Гра "Night in the Woods"

Головним персонажем у грі виступає антропоморфна кішка. Усі інші жителі міста також є антропоморфними тваринами, які живуть «людське» життя – ходять на роботу, мають дім, одяг.

Рисунок 1.3. – Дизайн персонажів гри " Night in the Woods"

Це пригодницька гра, гравці якої зосереджуються на дослідженні, дізнаються історії персонажів, яких зустрічають. Після успішного кікстартеру гру створює InfiniteFall, команда Алека Холовки (Aquaria), Скотта Бенсона (LateNightWorkClub) і Бетані Гокенберрі [5].

Управління у грі здійснюється за допомогою клавіатури та мишки, джойстика або дотиків до екрану, формат гри – 2d платформер. Платформи, на яких доступна гра: Nintendo Switch, PS4, Xbox One, macOs, iOS, Microsoft Windows, Linux, MacOS [6].

«RaymanRaving Rabbids». Опис гри: Натовпи божевільних кроликів, що вийшли з-під контролю, вторглися, поневоливши Реймана та змусивши його брати участь у випробуваннях у стилі «гладіатор» [7].

Raving Rabbids – серія ігор, які в 2006 році випустила компанія Ubisoft. Серед них і гра Rayman Raving Rabbids, яка розроблялась для консолей Xbox 360, NintendoWii, PlayStation 2, персональних комп'ютерів та мобільних телефонів. Оскільки гра була представлена в той же час що і запуск консолі NintendoWii розробники сконцентрувались на якості управління грою за допомогою WiiRemote [8].

Рисунок 1.4 – Гра «Raving Rabbids»

Усі персонажі «Raving Rabbids» виглядають однаково і мають антропоморфні риси (ходять на двох ногах, можуть брати предмети руками) і тваринну зовнішність (прототипом дизайну був кролик).

Dust: An Elysian Tail. Опис гри: Дія Dust відбувається у світі, населеному антропоморфними істотами, де головний герой, Даст, зустрічає розумний меч, Клинок Ахра, і його охоронця Непосиду. Не пам'ятаючи про своє минуле, Даст дотримується поради Фіджета та допомагає населенню світу протистояти військам під проводом генерала Гая. Це пригодницький бойовик із дослідженням у стилі Metroidvania, який вимагає від гравця посилень, щоб Даст міг досягти нових територій.

Dust: AnElysianTail – це рольова відеогра, розроблена американським незалежним дизайнером Діном Додрилом, видана Microsoft Studios. Він був випущений для Xbox 360 через XboxLiveArcade у серпні 2012 року, а згодом для Microsoft Windows у травні 2013 року, для Linux і OS X у грудні 2013 року та для PlayStation 4 у жовтні 2014 року. Версія для Nintendo Switch була анонсовано на виставці E3 2018 і випущено у вересні 2018 року [9].

Рисунок 1.5. – Гра «Dust: An Elysian Tail»

Дизайн головного персонажу ближчий до людини ніж в попередніх прикладах, але ми бачимо очевидно тваринні риси – вуха, хвіст, тваринну мордочку. Одяг та зброя персонажа нічим не відрізняються від людських.

SpyFox. Опис гри: SpyFox – це серія програмних ігор від Humongous Entertainment із вигаданою антропоморфною лисицею з однойменною назвою, призначена для дітей віком від 8 років.

В іграх також є бігові ґеги, як-от пояс Мавпи Пенні для карате, який з'являється на упаковці, але не відображається в реальному ігровому процесі (хоча пояс з'являється в бонусній кінцівці з OperationOzone і іноді показується в анімаціях, які відтворюються під час титрів) [10].

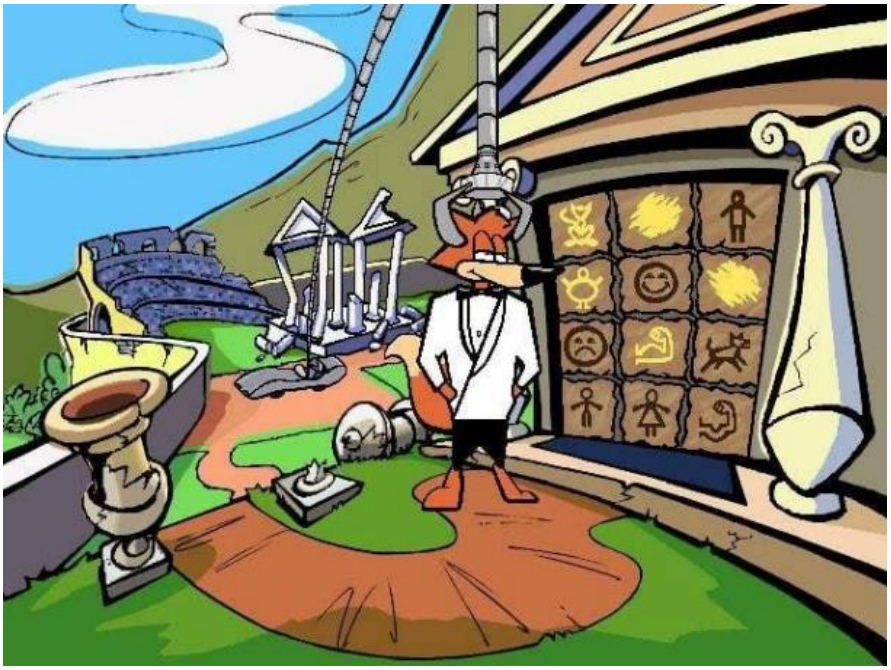

Рисунок 1.6. – Гра «SpyFox»

Багато назв гри таелементів сюжету є пародією на серіали про Джеймса Бонда та "Будь кмітливим".

Гра випущена на таких платформах: Macintosh, iOS, Linux, Android Windows [10].

1.6. Розподіл додатків за тематикою

За останні декілька років мобільні додатки стали невід’ємною частиною нашого життя. Ці додатки допомагають нам у вирішенні різних задач – від спрощення повсякденних рутинних справ до допомоги у складних проектах. Розподіл додатків за тематикою може бути корисним для визначення того, яку

екологічну тематику більше за все використовують у додатках та які типи додатків мають найбільший попит серед користувачів, які теми потребують більшої уваги.. Наприклад, можна знайти додатки з тематикою відновлення лісів або рекультивації земель, додатки для збору сміття або видалення небезпечних речовин з природи.

Групування мобільних додатків за їх призначенням можна зробити на різних рівнях деталізації. Багато компаній та користувачів звертають увагу на питання екології та стали активно займатися його вирішенням. У цьому контексті екологічні мобільні додатки стали дуже популярними. Розглянемо основні групи екологічних додатків.

Додатки для вимірювання викидів – ця група додатків дозволяє вимірювати викиди транспорту, побутових приладів, а також відраховувати кількість вуглецю, який споживається для виготовлення різних продуктів. Найбільш популярними додатками в цій групі є Carbon Footprint, MyClimate і Carbon Tracker.

Додатки для екологічних покупок – додатки цієї групи дозволяють користувачам здійснювати екологічні покупки, враховуючи питання збереження довкілля. Вони надають інформацію про екологічність товарів, що дозволяє користувачам зробити свідомий вибір. До цієї групи належать додатки, такі як Good On You, Ecosia та HappyCow.

Додатки для збереження води та енергії – ця група додатків дозволяє користувачам зберігати енергію та воду, що робить їх більш екологічними. Додатки, такі як WaterMinder, JouleBug та WattTime, стали дуже популярними в цій групі.

Додатки для відслідковування екологічних проблем – ця група додатків дозволяє користувачам відстежувати екологічні проблеми, які виникають у їх регіоні або в інших місцях світу. Ці додатки допомагають зрозуміти, які екологічні проблеми потребують найбільшої уваги. До цієї групи належать додатки, такі як Earth Now, Eco Watch та WWF Together.

Розподіл мобільних додатків за тематикою дозволяє зрозуміти, які типи додатків мають найбільший попит серед користувачів. Екологічні додатки допомагають користувачам зрозуміти питання екології та стати більш свідомими у виборі своїх дій та покупок. Додатки для вимірювання викидів, екологічних покупок, збереження води та енергії та відслідковування екологічних проблем - це основні групи екологічних додатків, які користуються популярністю серед користувачів. Крім того, екологічні додатки також допомагають зменшити вплив людей на довкілля та забезпечити сталість нашого планети.

Зараз, коли все більше людей усвідомлюють потребу у збереженні довкілля, екологічні додатки стають все популярнішими. Це пов'язано з ростом інтересу до сталих технологій та збереження навколишнього середовища. У нашому часі стає все важливішим враховувати вплив наших дій на довкілля та діяти відповідально. Тому, екологічні додатки можуть стати потужним інструментом для збереження довкілля та забезпечення сталості нашого планети.

У заключенні, можна сказати, що розподіл мобільних додатків за тематикою є дуже важливим для зрозуміння потреб користувачів. Екологічні додатки стають все популярнішими серед користувачів, що свідчить про те, що люди починають усвідомлювати важливість сталості та збереження довкілля. Загалом, екологічні додатки допомагають зробити світ кращим для нас і майбутніх поколінь.

1.7. Розподіл додатків за платформами.

У сучасному світі, де мобільні пристрої стали необхідністю для більшості людей, розробка мобільних додатків є важливою складовою розвитку бізнесу. Однак, вибір правильної платформи для розробки може

стати вирішальним фактором у досягненні успіху у даній галузі. Далі будуть розглянуті основні платформи для розробки мобільних додатків та їх переваги та недоліки.

Android є найпопулярнішою платформою для розробки мобільних додатків з більш ніж 85% ринку мобільних пристроїв, що працюють під управлінням цієї ОС. Однією з головних переваг Android є широкий охоплюваний ринок, що робить його привабливим для розробників додатків. Крім того, Android є відкритою платформою, що дозволяє розробникам модифікувати його за потребою та розширювати можливості стандартного SDK. Проте, на Android можуть виникнути проблеми зі сумісністю додатків з різними версіями ОС та різними розмірами екрану, що може призвести до проблем зі стабільністю та роботою додатків. Також, додатки на Android можуть бути менш безпечними, оскільки платформа дозволяє завантажувати додатки з будь-яких джерел, що може відкривати доступ до шкідливого програмного забезпечення.

iOS є другою за популярністю платформою для розробки мобільних додатків та має більш ніж 15% ринку мобільних пристроїв. Однією з головних переваг iOS є її екосистема, яка дозволяє розробникам додатків швидко та легко публікувати свої додатки в App Store та забезпечує більш високу стійкість та безпеку додатків. Крім того, розробка додатків для iOS зазвичай є менш складною, оскільки для цього використовується одна версія ОС та обмежений набір розмірів екрану. Однак, iOS також має свої недоліки, зокрема високі вимоги до затвердження додатків для публікації в App Store та обмежену свободу для розробників у модифікації ОС та розширенні функціональності. Крім того, ціни на пристрої з iOS можуть бути вищими, що може обмежувати ринок користувачів.

Hybrid – це платформа для розробки мобільних додатків, що поєднує можливості розробки веб-додатків з можливостями мобільних додатків. Hybrid дозволяє розробникам створювати додатки з використанням HTML,

CSS та JavaScript, що дозволяє швидко розробляти та випускати додатки на різних платформах. Однією з головних переваг Hybrid є можливість зберегти час та кошти на розробці додатків для різних платформ, оскільки додатки можуть бути випущені на різних платформах з використанням одного коду. Крім того, Hybrid дозволяє розробникам створювати додатки з використанням відкритих стандартів, що робить їх більш безпечними. Однак, Hybrid має свої недоліки, зокрема обмежені можливості розширення функціональності та графічного дизайну додатків, що може обмежувати їх можливості порівняно з нативними додатками. Крім того, Hybrid-додатки можуть бути менш ефективними, оскільки вони запускаються в WebView, що може призводити до погіршення продуктивності та швидкодії.

React Native – фреймворк, який використовується для розробки мобільних додатків, що базується на JavaScript та дозволяє розробникам створювати мобільні додатки з використанням відкритих стандартів. React Native використовує компоненти та бібліотеки, що дозволяє розробникам створювати мобільні додатки, що мають нативний вигляд та функціональність. Однією з головних переваг React Native є можливість швидкої розробки та випуску мобільних додатків для різних платформ з використанням одного коду. Крім того, React Native має високу продуктивність та швидкодію, що дозволяє розробникам створювати додатки, які працюють швидко та ефективно. Однак, React Native має свої недоліки, зокрема обмежену можливість розширення функціональності та графічного дизайну додатків. Крім того, React Native може бути складнішим для вивчення, особливо для розробників, які не мають досвіду з React [11].

Вибір платформи для розробки мобільних додатків залежить від багатьох факторів, включаючи цільову аудиторію, потреби користувачів, бюджет та технічні можливості. Нативні додатки зазвичай забезпечують кращу продуктивність та функціональність, але можуть вимагати більше часу та коштів на розробку та тестування. Hybrid-додатки можуть бути більш

ефективними з точки зору швидкості розробки та менш витратними, але можуть мати обмежені можливості та нижчу продуктивність.

При виборі платформи для розробки мобільних додатків рекомендується враховувати такі фактори, як цільова аудиторія, функціональні вимоги, технічні можливості та бюджет. Найкращим варіантом може бути розробка нативних додатків для кожної платформи, яка підтримується, якщо дозволяє бюджет та час на розробку.

У кінці кінців, вибір платформи для розробки мобільних додатків залежить від конкретних потреб та обставин. Важливо провести достатньо часу на дослідження та порівняння різних варіантів та врахувати всі фактори, щоб забезпечити успішну розробку та випуск мобільного додатку.

Крім того, важливо також врахувати наявність підтримки та спільноти користувачів та розробників для обраної платформи. Наявність документації, онлайн-курсів, форумів та інших ресурсів можуть значно спростити розробку та допомогти вирішувати технічні питання. Також, важливо враховувати тенденції ринку та рівень конкуренції на ньому. Наприклад, якщо більшість конкурентів використовують нативні додатки, то використання гібридних або крос-платформенних рішень може знизити конкурентоспроможність додатку. У той же час, якщо ринок насичений нативними додатками, то гібридні або крос-платформенні додатки можуть стати ефективним варіантом для швидкої та економічної розробки.

Отже, вибір платформи для розробки мобільних додатків є важливим етапом у процесі створення успішного та популярного додатку. Для кожного проекту варто враховувати специфічні вимоги та обставини та провести аналіз та порівняння різних варіантів. Всі три варіанти, нативний, гібридний та крос-платформенний, мають свої переваги та недоліки, тому важливо зважити на кожен фактор перед прийняттям рішення. Також, важливо враховувати наявність підтримки та ресурсів для обраної платформи та тенденції на ринку.

1.8. Висновки за першим розділом

Екологічна тематика в мобільних додатках є досить актуальною темою, оскільки людство стикається зі зростаючими проблемами довкілля та змінами клімату. Реалізація антропоморфізму в мобільних додатках також має потенційні ризики в контексті стимулювання ілюзії емпатії з тваринами та змінення уявлень про природу та її значення.

Аналіз розподілу мобільних додатків за тематикою та платформами показав, що екологічні додатки на даний момент є не такими популярними, як, наприклад, ігрові додатки. Проте, це може свідчити про потенційну нішу на ринку мобільних додатків, яку можна заповнити, створивши більше додатків з екологічною тематикою.

Також, зазначено, що дослідження тенденцій у розподілі мобільних додатків за платформами може бути корисним для визначення переваг та недоліків кожної з платформ, а також для розробки оптимальної стратегії розробки мобільних додатків з екологічною тематикою.

При написанні роботи були досліджені методи розробки ігор для різних платформ, також розглянуто версії, що працюють без використання графіки. Враховувалась можливість розміщення, просування та підтримки гри на різних платформах (PlayMarket, AppStore, Steam). Також, оскільки гра має бути доступною для якомога більшої кількості людей, додаток не повинен мати високих програмних вимог.

В результаті для розробки додатку було обрано Android-платформу. Це дозволяє охопити максимальну кількість людей, адже зараз більшість користувачів в Україні використовують сааме операційну систему Android [12].

У подальшому дослідженні буде розглянуто конкретний приклад мобільного додатку з екологічною тематикою, його особливості та внесок у розв'язання екологічних проблем.

РОЗДІЛ 2
ІНСТРУМЕНТАЛЬНІ ЗАСОБИ СТВОРЕННЯ МОБІЛЬНОГО ДОДАТКУ

2.1. Вимоги до апаратних платформ на сучасному етапі

Вимоги до апаратних платформ на сучасному етапі зростають зі збільшенням складності завдань, які ставляться перед комп'ютерною технікою. Використання комп'ютерної техніки нині стало невід'ємною складовою нашого життя, а вимоги до апаратних платформ наразі є дуже високими. Апаратні платформи – це складні системи, які складаються з різноманітних компонентів, таких як процесор, оперативна пам'ять, накопичувач, материнська плата, відеоприскорювач тощо. Якість та продуктивність роботи комп'ютера залежить від цих компонентів.

Сучасні апаратні платформи повинні відповідати деяким вимогам. З одного боку, вимоги до апаратних платформ визначаються потребами користувачів, таких як геймери, музиканти, графічні дизайнери, науковці та інші. Для геймерів і музикантів важливою є якість звуку та зображення, а також швидкість обробки даних для запуску вимогливих ігор і програм. Для графічних дизайнерів і науковців важливою є швидкість обробки графіки та великих обсягів даних, які використовуються для створення складних зображень та виконання наукових розрахунків.

З іншого боку, розвиток Інтернету та інформаційних технологій призводить до зростання кількості підключених до мережі пристроїв. Тому важливою вимогою є забезпечення мережевої сумісності та можливості працювати в різних середовищах.

Іншими важливими вимогами є забезпечення безпеки, стійкості та надійності апаратних платформ. Застосування уразливих апаратних платформ може призвести до крадіжок конфіденційної інформації, атак на комп'ютерні

системи та інші загрози безпеці. Окрім того, важливою є енергоефективність апаратних платформ, оскільки вона забезпечує зменшення споживання енергії та зменшення витрат на електроенергію.

Також вимоги до апаратних платформ пов'язані з екологічними проблемами. Виробництво та експлуатація комп'ютерної техніки може мати негативний вплив на навколишнє середовище, тому важливо, щоб апаратна платформа була виготовлена з використанням екологічно чистих матеріалів та мала низький рівень енергоспоживання.

У майбутньому, з розвитком технологій, можуть з'явитися нові вимоги до апаратних платформ, такі як здатність працювати зі штучним інтелектом, розпізнаванням обличчя та іншими новими технологіями. Ці нові вимоги, разом з вже існуючими, будуть визначати напрямок розвитку апаратної частини комп'ютерних систем у майбутньому.

Сучасні технології розвиваються настільки швидко, що мобільні пристрої стали одним з класів техніки, які найбільш інтенсивно розвиваються. Їх апаратні можливості вже повністю відповідають можливостям традиційних ПК, але це не означає, що вимоги до них не відрізняються. Загальні вимоги до мобільних платформ включають обчислювальну потужність, енергоефективність, сумісність, інформаційну безпеку, надійність та стабільність роботи. Таким чином на сьогоднішній день мобільні пристрої мають вражаючі можливості, що можуть забезпечити їхню конкурентоспроможність на ринку технологій. До того ж вони частіше використовуються в буденному житті та більш мобільні. Отже, при роботі над дипломним проєктом було обрано мобільні пристрої на платформі Android.

2.2. Програмні середовища для створення Android-додатків

Програмні середовища для створення Android-додатків є необхідним

інструментом для розробки мобільних додатків для операційної системи Android. Ці середовища дозволяють розробникам створювати додатки за допомогою графічних інтерфейсів, вбудованих компонентів та інструментів, що значно спрощує процес розробки та забезпечує високу якість готового продукту.

Один з найпопулярніших програмних інструментів для створення Android-додатків є Android Studio. Це інтегроване середовище розробки, розроблене компанією Google, яке має широкий набір інструментів та функцій. Android Studio підтримує розробку на мові програмування Java, C++ та Kotlin, має інтегровану систему контролю версій та надає доступ до великої кількості додаткових бібліотек та інструментів.

Іншим популярним інструментом є Unity, який використовується для створення ігрових додатків на Android. Unity надає можливість розробникам створювати складні ігрові сценарії та інтерактивні елементи за допомогою графічного інтерфейсу та скриптів на мові програмування C#.

Крім того, є й інші інструменти, такі як Xamarin та Flutter, які дозволяють розробляти кросплатформні додатки на Android та інші операційні системи. Xamarin використовує мову програмування C# та має велику кількість готових компонентів та бібліотек для розробки додатків, а Flutter базується на мові програмування Dart та надає широкі можливості для створення високоякісних та швидких додатків на різних платформах.

Однією з особливостей програмних середовищ для створення Android-додатків є можливість використання різних інструментів та технологій для розробки додатків, що дозволяє розробникам обирати оптимальний підхід для своїх проєктів. Наприклад, для розробки додатків, що вимагають великої швидкодії та точності, можна використовувати мову програмування C++, що дозволяє досягати високої продуктивності.

Крім того, програмні середовища для створення Android-додатків надають можливість розробникам тестувати додатки на емуляторі, що

дозволяє швидко та ефективно перевіряти роботу додатка на різних версіях операційної системи Android та різних типах пристроїв.

Також важливим аспектом при розробці додатків є можливість використовувати відкриті стандарти та протоколи, що дозволяє забезпечити відкритість та сумісність додатків з іншими програмними засобами та пристроями.

Однак, розробка додатків на Android має свої виклики та обмеження, такі як різноманітність пристроїв та їх особливостей, що вимагає врахування різних факторів при розробці додатку. Крім того, потрібно враховувати питання безпеки та конфіденційності при зборі та обробці даних користувачів.

У зв'язку зі швидким розвитком мобільних технологій, програмні середовища для розробки Android-додатків постійно оновлюються та покращуються. З'являються нові інструменти для автоматизації процесу розробки та тестування додатків, що дозволяє знизити витрати часу та зусиль при розробці додатків.

Вибір середовища розробки мобільних додатків для Android є важливим етапом, що впливає на якість та ефективність розробки. Для вибору середовища розробки були враховані критерії, що наведено далі.

Функціональність: програмне середовище повинне мати достатній функціонал для розробки потрібного додатку. Для різних проектів можуть вимагатися різні функції та інструменти, тому важливо вибрати середовище, яке надає необхідні можливості.

Зручність використання: програмне середовище повинне бути зручним та простим у використанні для розробника. Важливо, щоб інтерфейс середовища був інтуїтивно зрозумілим та легким для вивчення.

Підтримка: важливо, щоб програмне середовище мало достатню підтримку та поновлення від розробника. Наявність документації, інструкцій, відеоуроків, форумів та спільноти розробників може значно полегшити процес розробки.

Вартість: вартість програмного середовища також є важливим фактором. Деякі середовища можуть бути безкоштовними, а інші можуть вимагати плату за використання. Важливо розглянути бюджет проекту та визначити оптимальні витрати на розробку.

Підтримка мов програмування: програмне середовище повинне підтримувати мови програмування, які використовуються у проекті. Деякі середовища підтримують більше мов програмування, ніж інші, що може бути важливим фактором при виборі.

Наявність інструментів для тестування: важливо, щоб програмне середовище надавало інструменти для тестування додатку. Це дозволяє розробникам перевірити функціональність та якість додатку перед випуском.

Наявність інструментів для розгортання та збірки додатку: програмне середовище повинне надавати інструменти для розгортання та збірки додатку. Це дозволяє розробникам швидко та ефективно випускати нові версії додатку.

Наявність додаткових інструментів: програмне середовище може надавати додаткові інструменти, такі як плагіни, бібліотеки, шаблони та інші. Вони можуть значно полегшити процес розробки та забезпечити якість додатку.

Серед найбільш популярних середовищ розробки виділяють Xamarin, Flutter, IntelliJ IDEA, B4A, Eclipse, Android Studio, Unity.

Xamarin – це платформа з відкритим вихідним кодом для створення програм під iOS, Android і Windows за допомогою .NET. Xamarin керує зв’язком спільного коду з базовим кодом платформи та працює за допомогою керованого середовища, яке забезпечує, наприклад, розподіл пам’яті. Він дозволяє розробникам спільно використовувати в середньому 90% своїх програм на різних платформах [13]. Використовується для розробки додатків мовою C#.

Flutter — це фреймворк, розроблений компанією Google. Використовується для створення нативно скомпільованих

мультиплатформнних додатків. Flutter має відкритий код та написаний мовою Dart (ela.kpi.ua) [14].

Android Studio — IDE для розробки програм для Android. Базуючись на потужному редакторі коду та інструментах розробника від IntelliJ IDEA, Android Studio пропонує велику кількість функцій, які підвищують якість та продуктивність продукту.

У порівнянні Unity з Android Studio, Unity надає ширший спектр можливостей для розробки ігор та візуалізацій. Середовища, що порівнюються мають власний інтерфейс та функціональність. Однак, для розробників з досвідом у графічному дизайні та ігровій розробці, Unity може бути зручнішим використовувати. Обидва середовища є безкоштовними для використання та мають достатню підтримку від розробників та спільнот. Але у зв'язку з тим, що Unity є популярнішим середовищем для ігрової розробки, його спільнота активніша. Unity підтримує декілька мов програмування, включаючи C# та JavaScript, тоді як Android Studio підтримує тільки Java, C++ та Kotlin. Unity має більш розвинені інструменти для тестування та налагодження додатків, такі як Unity Test Runner та Unity Profiler. Окрім того, Unity надає можливість швидкого розгортання додатків на різних платформах та має ширший вибір плагінів та бібліотек.

Отже, на підставі вищенаведених критеріїв можна зробити висновок, що Unity має переваги порівняно з Android Studio в розробці ігор та інтерактивних додатків.

2.3. Платформа Unity

Unity – це одна з найпопулярніших платформ. Найчастіше використовується для створення 3d і 2d ігор, а також різних візуальних додатків. Unity дозволяє створювати ігри під різні платформи, наприклад,

мобільні пристрої, ПК, ігрові консолі, віртуальні та доповнені реальності. Ця платформа революціонізувала процес розробки ігор, роблячи його доступнішим та простішим для великої кількості розробників.

2.3.1. Основні відомості про платформу

Unity – це кросплатформний ігровий рушій. Компанія Unity Technologies вперше анонсувала його вихід у червні 2005 року на всесвітній конференції розробників від Apple як ігровий рушій для MacOSX. Відтоді він поступово розширювався для підтримки різноманітних комп'ютерів, мобільних телефонів, консолей і платформ віртуальної реальності. Він особливо популярний для розробки мобільних ігор для iOS та Android, вважається простим у використанні для розробників-початківців і популярний для розробки незалежних ігор [15]. Unity була написана мовою C#, вихідний код був опублікований у 2018 році [16]. Редактор Unity підтримують такі платформи як macOS, Windows і Linux, а сам механізм наразі підтримує створення ігор для понад 19 різних платформ. Unity має безкоштовні та платні варіанти ліцензування [15].

Unity може легко інтегруватися з Android, завдяки чому можна створювати ігри, що сумісні з переважною більшістю Android-пристроїв. Вона має вбудовану підтримку Android та може створювати або експортувати проект в APK-файл, що дозволяє легко розповсюджувати гру в Google Play або інші магазини додатків.

Ця платформа дозволяє легко керувати різними функціями пристрою Android, такими як геолокація, графіка, звук, мережа та інші, та має для цього багато вбудованих компонентів та інструментів, які дозволяють легко налаштувати та змінювати різні аспекти гри під час взаємодії з пристроєм Android.

Unity має велику спільноту розробників, яка допомагає отримувати відповіді на свої запитання та розв'язувати проблеми, що виникають під час створення гри. У цій спільноті зібрано велику кількість ресурсів та інформації, яка може допомогти у вирішенні проблем.

Вбудований рушій рендерингу Unity дозволяє створювати різноманітні графічні ефекти, такі як світло, тіні тощо, та візуальні елементи, що дозволяє створювати більш реалістичні та захоплюючі ігри.

Unity також має потужний звуковий рушій, який дозволяє створювати вражаючий звуковий дизайн для гри. Це дає змогу створювати реалістичні звуки та музику, що додає більшої атмосферності та глибини грі.

2.3.2. Технологія створення додатків в середовищі Unity

Для створення програм в середовищі розробки Unity існують шаблони. На початку роботи, при створенні проєкту, в UnityHub можна вибрати шаблон та завантажити його, а також встановити необхідні плагіни.

Під час розробки додатку є можливість миттєво переглянути вміст сцени за допомогою симулятора. Також є можливість вибрати необхідний девайс для симуляції – в цій роботі в симуляторі використовувались різні мобільні телефони для розрахунку розмірів вікон та відслідковування їх коректного відображення.

На сцені розміщуються усі необхідні об’єкти, їм надаються певні характеристики, такі як розмір, колір, зображення, скрипт для виконання, кнопки. До кожного об’єкту можна прикріпити скрипт – програмний код, написаний мовою C#.

Тестування додатку можливо в самій Unity, за допомогою Unity Play Mode. Коли додаток завершено та протестовано створюється збірка гри яка має назву Build. При створені збірки в першу чергу обирається платформа для

якої створюється збірка, далі редагуються відомості про гру – автор, назва додатку, версія, іконка.

2.4. Середовище розробки Microsoft VS

Microsoft Visual Studio — (elartu.tntu.edu.ua) IDE, створене компанією Microsoft. Ця середа розробки використовується для розробки різних комп'ютерних програм, включаючи веб-сайти, веб-програми, веб-сервіси та мобільні програми. VS дає змогу використовувати Windows API , Windows Forms, Windows Presentation Foundation, Windows Store (elartu.tntu.edu.ua) і Microsoft Silverlight. VS підтримує понад 35 мов програмування, окрім цього редактор коду та налагоджувач можуть підтримувати інші мови програмування, за умови що існує спеціальна служба для цієї мови. Серед вбудованих мов можна виділити C, C#, C++, C++, Visual Basic, .NET, F#, JavaScript, TypeScript, XML, XSLT, HTML і CSS. За допомогою плагінів також є можливість підтримки таких мов як Python, Ruby, (ela.kpi.ua) Node.js, і M [17].

2.5. Обґрунтування вибору мови програмування Unity

В середовищі розробки Unity у розробників є можливість використовувати такі мови програмування як UnityScript, C# або Boo [15]. Однак, від версії Unity 2018.1 UnityScript більше не підтримується, і рекомендується використовувати C# як основну мову розробки в Unity.

C# це об’єктно-орієнтовна мова програмування. Вона досить поширена в галузі розробки ігор, тому однією з переваг є великий обсяг документації, підручників та статей а також чисельну спільноту розробників. Unity надає

повну підтримку для C# і пропонує широкий набір інструментів, бібліотек і документації для розробки цією мовою. C# надає прямий доступ до Unity API, що дозволяє зручно використовувати всі можливості движка Unity для розробки гри. Це дає змогу легко маніпулювати об'єктами, скриптами, фізикою, анімацією та іншими компонентами гри.

2.6. Висновки за другим розділом

На сучасному етапі розвитку можна виділити клас техніки, що розвивається найбільш інтенсивними темпами – мобільні пристрої. Апаратні можливості цих пристроїв вже повністю відповідають можливостям традиційних ПК, тому загальні вимоги до мобільних платформ нічим не відрізняються, тобто вони пов'язані з обчислювальною потужністю, енергоефективністю, сумісністю, інформаційною безпекою, надійністю та стабільністю роботи.

Середовища розробки для мобільних пристроїв дещо відрізняються від засобів розробки для ПК, але в кінцевому результаті вибір оптимального середовища та підходу для розробки додатку, як звичайно, залежить від конкретних вимог проекту та потреб користувачів. Найкраще середовище для розробки додатку повинно забезпечувати зручний інтерфейс, ефективний редактор коду, багатий набір інструментів та можливість тестування додатку на різних пристроях.

Для розробки мобільних додатків існує багато середовищ, наприклад Unity, Android Studio, Xamarin, Flutter, IntelliJ IDEA, B4A, Eclipse, тощо. Зважаючи на вибір операційної системи, для якої буде створюватися додаток, а також приймаючи до уваги легкість освоєння, кількість бібліотек та доповнень, які створили користувачі, із перелічених для розробки середовищ було обрано Unity. Це потужна та ефективна платформа, шо дозволяє

створювати якісні сучасні додатки. На даному етапі Unity вважається однією з найкращих платформ для створення ігор на Android.

РОЗДІЛ 3
МОБІЛЬНИЙ ДОДАТОК «N_DANGERED»

3.1. Концепт гри

Для назви мобільного додатку було використано слово «endangered» (з англійської – під загрозою зникнення).

Добавлено примечание ([22]): Не треба це виділяти в окремий пункт, бо він неприпустимо малий.

Гра полягає в проходжені рівнів, отриманні ігрових персонажів та предметів для персонажів. Персонажі мають такі характеристики як здоров'я, сила атаки, здатність відбити атаку супротивника, точність удару. Деякі персонажі мають здатність до лікування себе та своєї команди. Механіка проходження рівнів реалізується у вигляді покрокового бою. Рівень вважається пройденим, якщо усі персонажі супротивника мають 0 одиниць здоров'я, і хоча б один персонаж з команди користувача має 1 чи більше одиниць здоров'я.

Для комфортної гри користувачі мають забезпечити персонажів необхідними для життя умовами, якщо умови не дотримані або дотримані не повністю – ігровий персонаж стає слабким або помирає(користувач втрачає можливість використовувати цього персонажа у грі). Забезпечення персонажів умовами для комфортного існування полягає в підборі та покупці певних предметів для кожного персонажа. Це сприяє розповсюдженню інформації про необхідні умови для різних тварин серед користувачів.

Також у грі використані олюднення екологічних катастроф, катаклізмів та забрудень – у вигляді супротивників, з якими користувачу доведеться боротись. При підборі команди треба враховувати наскільки ті чи інші тварини/риби/птахи, що є прототипами персонажів, зазнають шкоди від конкретного забруднення.

3.2. Графічний інтерфейс

На рисунку 3.1 зображено головний екран, на ньому присутня панель яка відображає кількість ігрової валюти: енергії, необхідної для проходження рівнів; монет, необхідних для покупки ігрових товарів в магазині; кристалів, необхідних для покупки нових персонажів. По натисканню на будь-яку валюту відкривається вікно магазину.

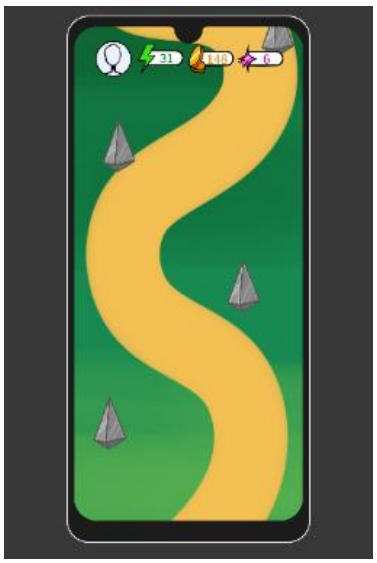

Рисунок 3.1 – Головний екран

Для вибору рівня необхідно взаємодіяти з кристалами, після натискання на нього він змінює колір, як зображено на рисунку 3.2. Надалі при вдосконаленні додатка при натискання на кристал буде відображатись меню для вибору персонажів та проходження рівня.

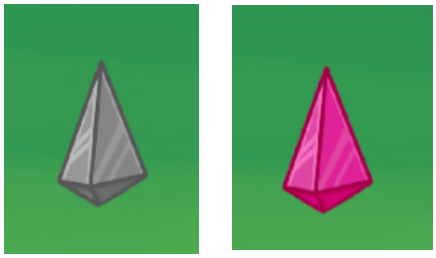

Рисунок 3.2 – Активація вибору рівня

Для переходу в налаштування, магазин або галерею було створено меню, зображене на рисунку 3.3, яке відкривається якщо натиснути на іконку користувача зліва вгорі.

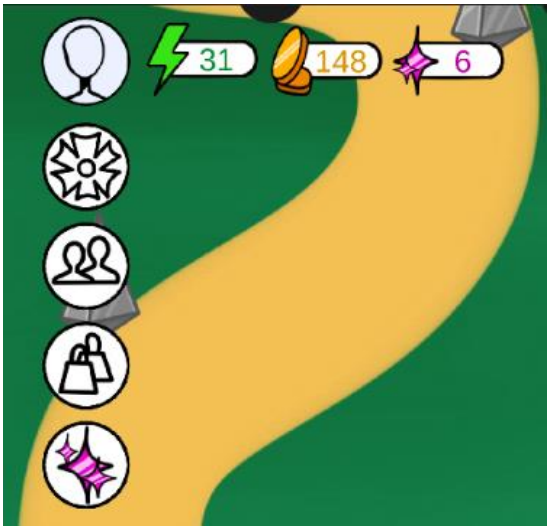

Рисунок 3.3 – Меню

Вікно налаштувань зображене на рисунку 3.4. При натисканні на кнопки «Звук» або «Музика» вимикається/вмикається звук або музика відповідно до попередніх налаштувань. Також змінюються іконки звуку та музики для того аби зробити інтерфейс більш зрозумілим для користувача.

Рисунок 3.4 – Налаштування

Для зміни мови виводиться окреме вікно, зображене на рисунку 3.5, з списком мов, на які можна змінити інтерфейс та кнопкою «Зберегти». Якщо користувач не вибрав жодної мови і натискає на кнопку «Зберегти» - інтерфейс залишитись на тій мові яка була до натискання на кнопку.

Рисунок 3.5 – Вікно вибору мови

При натисканні кнопки «Перезапуск» в налаштуваннях користувачу виводиться вікно з підтвердженням дії, зображене на рисунку 3.6. Якщо користувач натискає «Ні» вікно закривається і нічого не змінюються, якщо натискає «Так» – вікно налаштувань закривається, значення валюти гравця змінюється на «0», а також зникають усі персонажі і предмети які були куплені

під час гри.

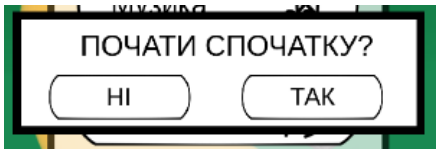

Рисунок 3.6 – Підтвердження перезапуску

В галереї персонажів, яку зображено на рисунку 3.7, представлено список персонажів які має гравець, назви тварин які використані для створення антропоморфних персонажів, а також рідкість персонажа, що позначається кількістю зірок справа зверху іконки персонажа.

Рисунок 3.7 – Галерея персонажів

На рисунку 3.8 зображено вікно «Магазин». Є можливість відсортувати товар по категоріям. Для покупки товару необхідно вибрати його та натиснути

кнопку «Купити» внизу екрану. При натисканні на будь-який товар внизу екрану змінюються деталі про покупку.

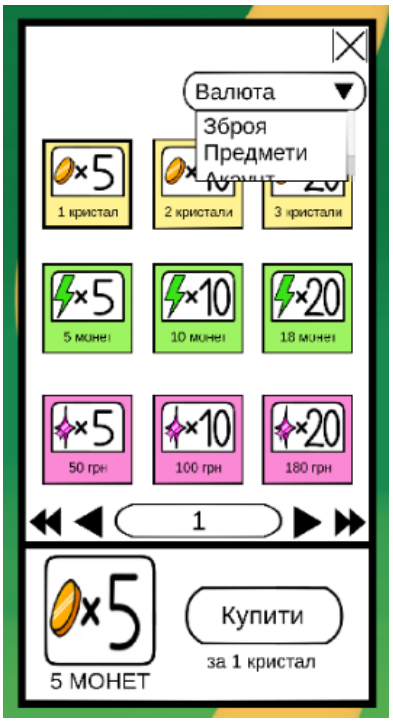

Рисунок 3.8 – Магазин

Для покупки персонажів необхідно вибрати останній пункт меню, відкриється нове вікно, зображене на рисунку 3.10. Є можливість повернутись назад за допомогою кнопки «Повернутись» та можливість отримати одного випадкового персонажа, натиснувши кнопку «Покликати».

Рисунок 3.10 – Вікно з покупкою персонажа

Рисунок 3.11 – Малюнок персонажа

Після покупки персонажа з’явиться вікно, зображене на рисунку 3.11, з малюнком персонажа та назвою. Для того аби закрити вікно та продовжити далі гру необхідно натиснути хрестик справа вгорі.

3.3. Програмна частина

Для роботи з об’єктами сцени за допомогою скриптів необхідно аби кожен клас скриптів наслідував клас Monobehaviour. Після цього посилання на будь-який об’єкт сцени можливий всередині Unity.

Об’єкти сцени мають компонент «Button», завдяки якому при натисканні на об'єкт запускаються функції скрипта, прив'язаного до цього об'єкту.

Змінні скриптів, які мають характеристику [SerializeField], з'являються в редакторі Unity і є змога змінювати їх або приписувати їм нові значення не редагуючи код. Саме це дозволяє приписувати певні об'єкти і компоненти в сцені до змінних в коді, щоб редагувати об'єкти не пов'язані з цим скриптом. Це дозволяє робити скрипти більш гнучкими та зменшити їх кількість. Для прикладу в скрипті«S_UI_ShowClose_window» створено дві змінні типу private, завдяки характеристиці SerializeField в редакторі Unity до цих змінних можна підв’язати будь-який об’єкт. При кожному новому використанні скрипта можна використовувати різні об’єкти.

Скрипт «S_UI_ShowClose_window»:

```
1.   public class S_UI_ShowClose_window : MonoBehaviour
2.   {
3.   [SerializeField] private GameObject window1;
4.   [SerializeField] private GameObject window2;
5.   public void ShowCloseWindow()
6.   {
7.   window1.SetActive(!window1.activeSelf);
8.   window2.SetActive(!window2.activeSelf);
9.   }
10.  }
```

Для використання сриптів необхідно додати до об'єкту компонент «Scripts» - це дає змогу використовувати усі публічні функції, записані в цьому скрипті, з використанням параметрів чи без. Це допомагає створити більш адаптивний і комфортний інтерфейс для користувача.

Для промотки пальцем всіх рівнів використовується вбудована мапа дій. InputActions, зображена на рисунку 3.12, яка повертає позицію в якій користувач доторкнувся екрану.

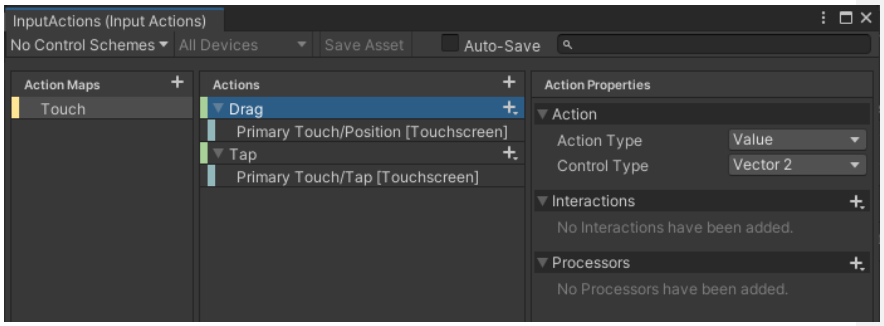

Рисунок 3.12 – Мапа дій, InputActions

Був створений об’єкт «input_manager», йому було надано два компоненти: Player Input та скрипт «S_TouchManager». Player Input необхідний для роботи з мапою дій. В скрипті «S_TouchManager» створено дві події – OnDrag(активується коли гравець торкається екрану) та OnDragCancelled(активується коли гравець відпускає палець і перестає торкатись екрану). В функції OnEnable() описана наступна логіка: якщо виконується подія дотику до екрану – запускається функція DragPerformed(), яка зчитує позицію дотику на початку і в кінці події. Це необхідно для розуміння яка саме дія була виконана гравцем – дотик чи гортання екрану. Якщо ж виконується відривання пальця від екрану викликається функція StopTouch(), яка запускає подію OnDragCancelled.

В класі S_Scroll створюється змінна типу S_TouchManager та об’єкт

який необхідно рухати під час гортання екрану.

В скрипті «Button_Audio» створено чотири публічні змінні: button_AudioSource, audio_Clip, touch_Manager, muted. За допомогою функцій OnEnable() та OnDisable() код запускає або зупиняє виконання події.

Код функції «Play_Sound()»:

```
1. public void Play_Sound()
2. {
3. button_AudioSource.PlayOneShot(audio_Clip);
4. }
```

Код функції «mute_sound()»:

```
5. public void mute_sound()
6. {
7. button_AudioSource.mute = muted;
8. muted = !muted;
9. }
```

Функція «Touch_Manager_OnTap()» викликає функцію «Play_Sound()», в якій реалізовано програвання звуку один раз по натисканню на кнопку. В функції «mute_sound()» реалізовано зміну характеристики об’єкту для заглушення звуку, перевірка чи звук був заглушений чи ні відбувається за допомогою відслідковування змінної «muted». По аналогії реалізована можливість вимикання або вмикання фонової музики.

Код класу «S_UI_choose_object»:

```
10.public class S_UI_choose_object : MonoBehaviour
11.{
12.public Sprite ActiveSprite, UnactiveSprite;
13.public Button fon_icon;
14.public Image icon;
15.public event EventHandler on_click;
16.public virtual void ChooseObject(int level)
17.{
18.on_click?.Invoke(this, EventArgs.Empty);
19.}
20.private void OnEnable()
21.{ fon_icon = GetComponent<Button>();
22.icon = GetComponent<Image>();
23.fon_icon.onClick.AddListener(() => { icon.sprite =
ActiveSprite; }); }
24.public virtual void UnChooseObject()
```

```
25.{
26.icon.sprite = UnactiveSprite;
27.}
28.}
```

В класі «S_UI_choose_object» реалізована зміна зображення об’єкта при натисканні на нього. Функція «ChooseObject» запускає подію натискання на об’єкт. Функція «OnEnable()» змінює зображення об’єкта по натисканню на нього.

Код класу «S_UI_choose_unit»:

```
29.public class S_UI_choose_unit : MonoBehaviour
30.{
31.public List<S_UI_choose_object> gameObjects = new
List<S_UI_choose_object>();
32.public virtual void Start()
33.{
34.foreach (S_UI_choose_object go in gameObjects)
35.{
36.go.on_click += S_UI_choose_object_on_click;
37.}
38.}
39.
40.public virtual void
S_UI_choose_object_on_click(object sender,
System.EventArgs e)
41.{
42.foreach (S_UI_choose_object go in gameObjects)
43.{
44.if (go != sender) go.UnChooseObject();
45.}
46.}
47.}
```

В класі «S_UI_choose_unit» створено лист куди поміщаються всі об’єкти доступні користувачу для вибору. При натисканні на будь-який із об’єктів запускається подія, в результаті якої вибраний об’єкт змінює зображення на активне, всі інші об’єкти автоматично змінюють зображення на неактивне.

За допомогою класів «S_UI_choose_object» та «S_UI_choose_unit» реалізовано зміну об’єктів при вибору рівня та товару в магазині.

3.4. Висновки за третім розділом

В результаті роботи був створений мобільний додаток для платформи Android. Реалізовані меню, налаштування, магазин, галерея персонажів, окреме вікно магазину з покупкою персонажів, екран з можливістю його промотки, відмітки рівнів з можливістю обрати необхідний рівень. Був вибраний оптимальний інструментарій для створення проєкту.

Додаток був спроєктований з урахуванням необхідного функціоналу, розроблений та реалізований алгоритм функціонування системи. Багаторазово протестовано та відлагоджено роботу додатка.

В розділі продемонстровано графічний інтерфейс, вказані можливості керування додатком.

В подальшому є можливість вдосконалити додаток:

1. Створити дизайн інших персонажів.
2. Створення ігрових рівнів.
3. Створення кросплатформних версій гри.
4. Вдосконалення механіки гри.
5. Вдосконалення інтерфейсу користувача.

ВИСНОВКИ

Під час написання цієї роботи було проаналізовано сучасний ринок ігор та мобільних додатків. Результати аналізу дають змогу стверджувати що на ринку невпинно зростає частка ігрового контенту, останнім часом користувачі з зацікавленістю зустрічають додатки з екологічною тематикою. Завдання створити додаток який охопить користувачів досить великого вікового діапазону та за його допомогою в ігровій формі просувати екологічну тематику, зокрема тему догляду за тваринами з Червоної книги та історія вимирання тварин з Чорної книги, є актуальним.

Данна робота базується на поєднанні мобільних додатків, які стають все більш поширеними навіть серед дітей та старшого покоління, та актуалізації екологічних проблем. В додатку використовуються антропоморфні персонажі, за прототип яких було взято тварин з Червоної та Чорної книг України. Це дає можливість утримати увагу користувача та надати новий досвід в процесі грі.

Для розробки використовувалась середа розробки Unity, редактору коду Microsoft Visual Studio та мова програмування C#.

Було створено мобільний додаток, який включає в себе меню, налаштування, ігровий магазин, галерею персонажів і можливість вибору рівня. Ця робота має перспективи для вдосконалення та створення гри, яка буде не тільки користуватися попитом на ринку а й відкриє можливості звернути увагу більшості людей на екологічні проблеми сьогодення.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Антропоморфізм. [Електронний ресурс] / Енциклопедія сучасної України. – Режим доступу: www. URL: https://esu.com.ua/article-43064
2. «Let's Talk About Furries». [Електронний ресурс]/Режим доступу: www. URL: https://oursaferschools.co.uk/2021/09/24/furries/
3. «Will Furries Ever Go Mainstream?». [Електронний ресурс]/Режим доступу: www. URL: https://www.rollingstone.com/culture/culture-features/furries-midwest-furfest-mainstream-932924/
4. Білоциц, П. Антропоморфізм як основний принцип створення рас у відеоіграх жанру fantasy. [Електронний ресурс] / П. Білоциц. Матеріали конференцій МНЛ, 23вересня 2022 р., м. Дніпро. 241–243 с. – Режим доступу: www. URL: https://archive.liga.science/index.php/conference-proceedings/article/view/121
5. Steam «Night in the Woods». [Електронний ресурс]/Режим доступу: www. URL: https://store.steampowered.com/app/481510/Night_in_the_Woods/
6. Night in the Woods. [Електронний ресурс]/Режим доступу: www. URL: https://en.wikipedia.org/wiki/Night_in_the_Woods
7. Rayman Raving Rabbids. [Електронний ресурс]/Режим доступу: www. URL: https://store.steampowered.com/app/15080/Rayman_Raving_Rabbids/
8. Rayman Raving Rabbids. [Електронний ресурс]/Режим доступу: www. URL: https://uk.wikipedia.org/wiki/Rayman_Raving_Rabbids
9. Dust: An Elysian Tail. [Електронний ресурс]/Режим доступу: www. URL: https://en.wikipedia.org/wiki/Dust:_An_Elysian_Tail
10. Spy Fox. [Електронний ресурс]/Режим доступу: www. URL: https://en.wikipedia.org/wiki/Spy_Fox
11. React.js Conf - A Deep Dive into React Native. [Електронний ресурс]/Режим доступу: www. URL: https://www.youtube.com/watch?v=7rDsRXj9-cU

12. Частка ринку Android та iOS: оприлюднено статистику 2022 року. [Електронний ресурс] / RootNation, Новини, Новини IT. – Режим доступу: www. URL: https://root-nation.com/ua/news-ua/it-news-ua/ua-android-ios-statistika-2022/
13. What is Xamarin? [Електронний ресурс]/Режим доступу: www. URL: https://learn.microsoft.com/en-us/xamarin/get-started/what-is-xamarin
14. Flutter Dev [Електронний ресурс]/Режим доступу: www. URL: https://flutter.dev/
15. Unity (game engine) [Електронний ресурс]/Режим доступу: www. URL: https://en.wikipedia.org/wiki/Unity_(game_engine)
16. Unity publishes reference-only C# source code on GitHub [Електронний ресурс]/Режим доступу: www. URL: https://www.pocketgamer.biz/news/67809/unity-publishes-reference-only-c-source-code-on-github/
17. Visual Studio [Електронний ресурс]/Режим доступу: www. URL: https://en.wikipedia.org/wiki/Visual_Studio

Додаток А

Код програми

Файл «Button_Audio.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class (unity3d.com) Button_Audio : MonoBehaviour
{
    public AudioSource button_AudioSource;
    public AudioClip audio_Clip;
    public S_TouchManager touch_Manager;
    public bool muted = true;
    private void OnEnable()
    {
        touch_Manager.OnTap += Touch_Manager_OnTap;
    }
    private void OnDisable()
    {
        touch_Manager.OnTap -= Touch_Manager_OnTap;
    }
    private void Touch_Manager_OnTap(object sender,
System.EventArgs e)
    {
        Play_Sound();
    }
    public void Play_Sound()
    {
        button_AudioSource.PlayOneShot(audio_Clip);
    }
    public void mute_sound()
    {
        button_AudioSource.mute = muted;
        muted = !muted;
    }
}
```

Файл «Music.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class (unity3d.com) Music : MonoBehaviour
{
    public bool muted = true;
    public AudioSource music_AudioSource;
```

```
    public void mute_sound()
    {
        music_AudioSource.mute = muted;
        muted = !muted;
    }
}
```

Файл «S_Scroll.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class (unity3d.com) S_Scroll : MonoBehaviour
{
    public S_TouchManager touchManager;
    public GameObject scrollGameObject;
    private float prevvY = -1;
    private bool isFirst = true;
    private void OnEnable()
    {
        touchManager.OnDrag += TouchManager_OnDrag;
        touchManager.OnDragCancelled +=
TouchManager_OnDragCancelled;
    }
    private void TouchManager_OnDragCancelled(object sender,
System.EventArgs e)
    {
        isFirst = true;
    }
    private void OnDisable()
    {
        touchManager.OnDrag -= TouchManager_OnDrag;
        touchManager.OnDragCancelled -=
TouchManager_OnDragCancelled;
    }
    private void TouchManager_OnDrag(object sender, Vector3 e)
    {
        if (!isFirst)
        {
scrollGameObject.GetComponent<RectTransform>().anchoredPosition
-= new Vector2(0, (prevvY - e.y) * 30);
        }
        else
        {
            isFirst = false;
        }
        Debug.Log(prevvY - e.y);
        prevvY = e.y;
    }
```

```
}
```

Файл «S_TouchManager.cs»

```
using System;
using UnityEngine;
using UnityEngine.InputSystem;
using UnityEngine.InputSystem.EnhancedTouch;

public class S_TouchManager : MonoBehaviour
{
    public event EventHandler<Vector3> OnDrag;
    public event EventHandler OnDragCancelled;
    public event EventHandler OnTap;
    private PlayerInput playerInput;
    private InputAction dragAction;
    private InputAction tapAction;
    private void Awake()
    {
        playerInput = GetComponent<PlayerInput>();
        dragAction = playerInput.actions["Drag"];
        tapAction = playerInput.actions["Tap"];
    }
    private void OnEnable()
    {
        EnhancedTouchSupport.Enable();
        dragAction.performed += DragPerformed;
        tapAction.performed += OnTapPerformed;
        UnityEngine.InputSystem.EnhancedTouch.Touch.onFingerUp
+= StopTouch;
    }
    private void OnDisable()
    {
        dragAction.performed -= DragPerformed;
        tapAction.performed -= OnTapPerformed;
        UnityEngine.InputSystem.EnhancedTouch.Touch.onFingerUp -
= StopTouch;
    }
    private void DragPerformed(InputAction.CallbackContext
context)
    {
        Vector3 position =
Camera.main.ScreenToWorldPoint(context.ReadValue<Vector2>());
        position.z = 0.0f;
        OnDrag?.Invoke(this, position);
    }
        private void StopTouch(Finger fin)
    {
        OnDragCancelled?.Invoke(this, EventArgs.Empty);
    }
```

```
    private void OnTapPerformed(InputAction.CallbackContext
context)
    {
        OnTap?.Invoke(this, EventArgs.Empty);
    }
}
```

Файл «S_UI_choose_object.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using System;

public class S_UI_choose_object : MonoBehaviour
{
    public Sprite ActiveSprite, UnactiveSprite;
    public Button fon_icon;
    public Image icon;
    public event EventHandler on_click;
    public virtual void ChooseObject(int level)
    {
        on_click?.Invoke(this, EventArgs.Empty);
    }
    private void OnEnable() (ela.kpi.ua)
    { fon_icon = GetComponent<Button>();
        icon = GetComponent<Image>();
        fon_icon.onClick.AddListener(() => { icon.sprite =
ActiveSprite; }); }
    public virtual void UnChooseObject()
    {
        icon.sprite = UnactiveSprite;
    }
}
```

Файл «S_UI_choose_unit.cs»

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;

public class (ela.kpi.ua) S_UI_choose_unit : MonoBehaviour
{
    public List<S_UI_choose_object> gameObjects = new
List<S_UI_choose_object>();
    public virtual void Start()
```

```
    {
        foreach (S_UI_choose_object go in gameObjects)
        {
            go.on_click += S_UI_choose_object_on_click;
        }
    }
    public virtual void S_UI_choose_object_on_click(object
sender, System.EventArgs e)
    {
        foreach (S_UI_choose_object go in gameObjects)
        {
            if (go != sender) go.UnChooseObject();
        }
    }
}
```

Файл «S_UI_close_window.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class (unity3d.com) S_UI_close_window : MonoBehaviour
{
    [SerializeField] private GameObject window;
    public void CloseWindow()
    {
        window.SetActive(!window.activeSelf);
    }
}
```

Файл «S_UI_settings_music.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class (blog.dicoding.com) S_UI_settings_music :
MonoBehaviour
{
    public Sprite ActiveSprite, UnactiveSprite;
    public Button setings_music;
    public Image icon;
    private void OnEnable() (ela.kpi.ua)
    {
        setings_music = GetComponent<Button>();
        setings_music.onClick.AddListener(() =>
        {
            if (IsEnable)
```

```
            {
                icon.sprite = ActiveSprite;
            }
            else
            {
                icon.sprite = UnactiveSprite;
            }
            IsEnable = !IsEnable;
        });
    }
    private bool IsEnable = false;
}
```

Файл «S_UI_settings_yes_restart.cs»

```
using System.Collections;
using System.Collections.Generic;
using TMPro;
using UnityEngine;
using UnityEngine.UI;

public class (ela.kpi.ua) S_UI_settings_yes_restart :
MonoBehaviour
{
    [SerializeField] private GameObject settings;
    [SerializeField] private GameObject check_restart;
    [SerializeField] private TextMeshProUGUI energy;
    [SerializeField] private TextMeshProUGUI money;
    [SerializeField] private TextMeshProUGUI crystals;
    public void Restart()
    {
        settings.SetActive(!settings.activeSelf);
        check_restart.SetActive(!check_restart.activeSelf);
        energy.text = "0";
        money.text = "0";
        crystals.text = "0";
    }
}
```

Файл «S_UI_show_window.cs»

```
using System.Collections;
using (ela.kpi.ua) System.Collections.Generic;
using UnityEngine;

public class (unity3d.com) S_UI_show_window : MonoBehaviour
{
    [SerializeField] private GameObject window;
   public void ShowWindow () { window.SetActive
(!window.activeSelf); }
```

```
}
```

Файл «S_UI_ShowClose_window.cs»

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class (unity3d.com) S_UI_ShowClose_window : MonoBehaviour
{
    [SerializeField] private GameObject window1;
    [SerializeField] private GameObject window2;
    public void ShowCloseWindow()
    {
        window1.SetActive(!window1.activeSelf);
        window2.SetActive(!window2.activeSelf);
    }
}
```