ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Пояснювальна записка

До бакалаврської роботи
На ступінь вищої освіти бакалавр

На тему: **«РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ЕЛЕМЕНТАМИ ІГРОВОГО НАВЧАННЯ АНГЛІЙСЬКОЇ МОВИ З ВИКОРИСТАННЯМ UNITY 2D ДЛЯ ПРОГРАМИ ANDROID»**

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

Напненко М.В.
(прізвище та ініціали)

Керівник Гаманюк І.М.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Київ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

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

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____Негоденко О.В.
"____" _____2021 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТА

НАПНЕНКУ МАКСИМУ ВІТАЛІЙОВИЧУ

(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка програмного забезпечення з елементами ігрового навчання англійської мови з використанням Unity2D для програми Android»

Керівник роботи: Гаманюк І.М., старший викладач кафедри ІПЗ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «12» березня 2021 року № 65.
2. Строк подання студентом роботи «01» червня 2021 року
3. Вхідні дані до роботи
Теоретичні відомості щодо розробки програмного забезпечення для мобільних платформ ;

4. Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).
4.1 Актуальність обраної теми.
4.2 Опис використаних інструментів для розробки.
4.3 Аналоги програмного продукту.
4.4 Проектування.

4.5 Розробка.
4.6 Тестування.

5. Перелік демонстраційного матеріалу (назва основних слайдів)
1. Мета, об’єкт та предмет дослідження.
2. Існуючі аналоги.
3. Програмні засоби реалізації.
4. Діаграми та класи.
5. Висновки.

6. Дата видачі завдання «19» жовтня 2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів	Примітка
1.	Ознайомлення з необхідними матеріалами та літературою	19.04 – 22.04	Виконано
2.	Встановлення вимог до системи	25.04 – 05.05	Виконано
3.	Розробка та тестування	06.05 – 15.05	Виконано
4.	Письмова частина	16.05 – 20.05	Виконано
5.	Здача роботи	01.06.2021	

Студент _____ Напненко М.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гаманюк І. М.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина робота містить 54 с., 7 табл., 19 рис., 2 дод., 13 джерел.

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ЕЛЕМЕНТАМИ ІГРОВОГО НАВЧАННЯ АНГЛІЙСЬКОЇ МОВИ З ВИКОРИСТАННЯМ UNITY2D ДЛЯ ПРОГРАМИ ANDROID.

Об'єкт дослідження – застосування програмного забезпечення для покращення вивчення англійської мови.

Предмет дослідження – програмний застосунок, що дає можливість покращити знання граматики англійської мови.

Мета дослідження – покращення процесу вивчення англійської мови шляхом створення програмного забезпечення.

Під час виконання дипломної роботи було проведено дослідження ринку мобільних застосунків. Проаналізовані існуючі аналоги програмного продукту, виявлені їх переваги та недоліки. В роботі наведений опис існуючих інструментів для розробки програмного забезпечення.

В результаті проведених досліджень було вирішено розробити застосунок використовуючи рушій «Unity» для платформи «Android». В якості мови програмування буде виступати «C#».

Крім цього, під час проектування, було виконано ряд завдань:

- Модель предметної галузі (діаграма предметної галузі);
- Модель прецедентів (набір прецедентів та діаграми варіантів використання);
- Модель проектування (діаграми послідовностей, класів, діяльності);
- Реалізовано програмне забезпечення (діаграма артефактів, (код у додатку)).

Даний застосунок може використовуватись учнями та студентами.

Ключові слова: C#, Unity UML, Android, Microsoft Visual Studio.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

UML – Unified Modeling Language. Система позначок для представлення структури програми та її аналізу.

Скрипт – сценарій, програма, яка автоматизує певну задачу.

HCD – Human-centered-design. Це метод розробки програмного забезпечення, який фокусується, в першу чергу, на вимогах користувача.

IT – Information Technology. Обширна галузь, яка являє собою набір процесів для обробки, передачі та аналізу інформації.

Кастомізація – це процес адаптації та налаштування певних властивостей продукту для задоволення потреб різних груп користувачів.

UWP – Universal Windows Platform. Платформа для розробки програм під операційну систему «Windows 10»

WPF – Windows Presentation Foundation. Використовується для створення графічних інтерфейсів.

ПЗ – Програмне Забезпечення.

ВСТУП

Об'єкт дослідження – застосування програмного забезпечення для покращення вивчення англійської мови.

Предмет дослідження – програмний застосунок, що дає можливість покращити знання граматики англійської мови.

Мета дослідження – покращення процесу вивчення англійської мови шляхом створення програмного забезпечення.

За останні роки тенденція переходу користувачів зі звичайних мобільних телефонів з малою кількістю функцій на багатофункціональні смартфони та планшети стрімко збільшується. Зараз важко уявити людину, яка не користується інтернетом, не має облікового запису в якійсь популярній соціальній мережі або не грає в мобільні ігри. Смартфон планшет чи смарт-годинник є невід'ємним атрибутом сучасної людини, вони дозволяють людині отримати доступ до мережі Інтернет майже в будь-якому місці, будь то автобусна зупинка чи лавка в парку. Через це використання мобільних пристроїв набуло великої популярності в усіх куточках світу.

Кожного дня людина в середньому витрачає 3 год. 40 хв. на використання мобільного телефону[2]. Виникає проблема браку часу для навчання. Ця проблема є особливо актуальною для студентів вищих навчальних закладів та підлітків, які навчаються в школах. Тому було вирішено розробити програму для полегшення процесу вивчення англійської мови в ігровій формі.

В процесі дослідження вирішувалися наступні завдання:

- Актуальність теми;
- Вибір інструментів для розробки;
- Огляд аналогів застосунку, який розробляється;
- Визначення вимог до системи;
- Опис процесу розробки.

Статистика використання мобільних девайсів за останній рік наведена на сайті «Звіт про інтернет дані»[2]. Більш детально популярність мобільних девайсів та опис процесу розробки мобільних застосунків наведені на сайті «Розробка мобільних застосунків»[12]. Необхідну інформацію для роботи з ігровим рушієм «Unity» та інсталяційний пакет було взято з офіційного сайту ігрового рушія[8]. Середовище розробки «Microsoft Visual Studio 2019» можна завантажити на сайті розробника[1]. UML-діаграми описані в документації[5].

Наукова новизна справжнього дослідження полягає у проектуванні та розробці програмного забезпечення для покращення процесу вивчення англійської мови, використанні статистичних даних, роботі з новими функціями рушія «Unity».

Практична значущість полягає у можливості використання застосунку для покращення знань з граматики англійської мови школярами та студентами вищих навчальних закладів.

В якості ігрового рушія для розробки було обрано «Unity». Застосунок буде написаний на мові програмування «C#». Для написання коду буде використане середовище розробки «Microsoft Visual Studio 2019».

1 РОЗРОБКА МОБІЛЬНИХ ЗАСТОСУНКІВ. ОГЛЯД АНАЛОГІВ

1.1 Розробка мобільних застосунків

Ринок мобільних застосунків стрімко розвивається. Потужність девайсів на базі операційних систем Android та iOS росте, тому можливості для застосування та розробки мобільного програмного забезпечення збільшуються. За даними порталу «DataReportal», станом на 2020-й рік (рис. 1.1), кількість користувачів мобільних пристроїв – 5.19 мільярдів, що становить 67% від населення планети, кількість інтернет користувачів – 4.54 мільярди, що становить 59% від населення планети, кількість користувачів соціальних мереж – 3.80 мільярди, що становить 49% від населення планети.[2]

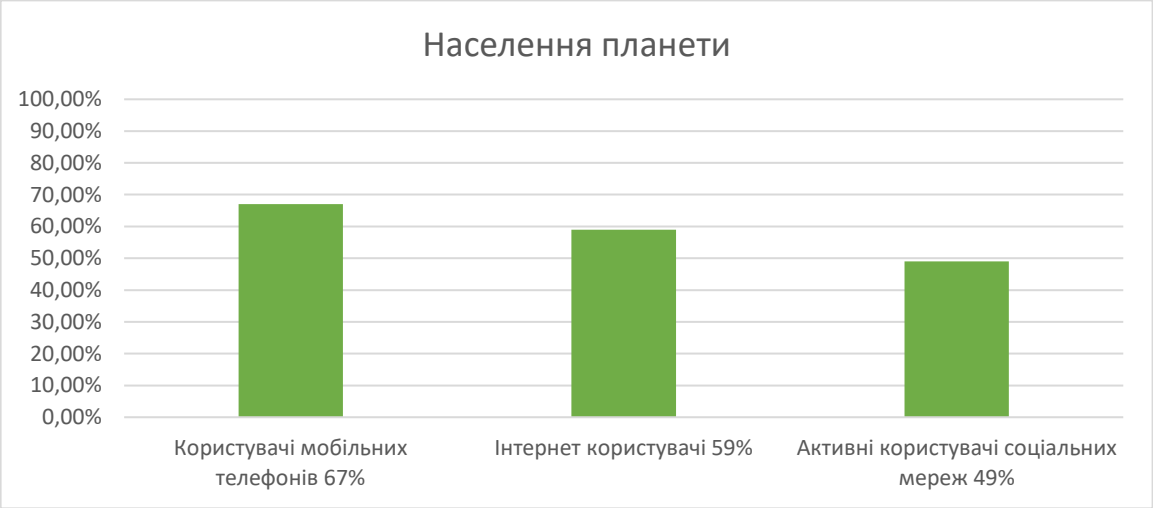


Рисунок 1.1 – Популярність цифрових технологій

Найбільша кількість завантажень та споживчих витрат користувачів Android та iOS приходиться на ігрові застосунки. Згідно статистики «The State of Mobile in 2020» мобільні ігри займають 56% ринку, а прибуток від ринку мобільних ігор за 2020-й рік склав 100 мільярдів доларів[7].

Тому розробка під мобільні платформи є дуже прибутковою і актуальною. Розробка мобільних застосунків являє собою процес написання коду, дизайн користувацького інтерфейсу для створення програмного забезпечення, яке буде використовуватись пристроями на базі мобільних операційних систем, таких як: iOS, Android, Windows Phone, Symbian, та подальшої підтримки застосунку після його публікації. В процесі розробки програмного забезпечення для мобільних пристроїв потрібно звернути увагу на деякі особливості:

- Різний форм-фактор пристроїв;
- Постійна взаємодія з Інтернетом;
- Адаптивність користувацького інтерфейсу;
- Різна потужність мобільних пристроїв.

Під час проектування мобільного застосунку потрібно орієнтуватися на принципи Human-centered-design (HCD).

Рисунок 1.2 – Етапи HCD

HCD – це спосіб проектування програмного забезпечення який орієнтований на користувача та його потреби. Принципи проектування HCD описані в міжнародному стандарті «(ISO 9241-210:2010 “Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems”)».[5]

Також, під час розробки програмного забезпечення важливо правильно підібрати інструменти які будуть використовуватись в процесі. Для розробки ігрового застосунку потрібно обрати рушій, на якому його буде створено. Серед безкоштовних рушіїв виділяють багатоплатформовий рушій Unity.

1.2 Ігровий рушій «Unity»

«Unity» – умовно безкоштовний, кросплатформний рушій для розробки ігрових програм. Починаючи з версії «Unity 2018.1.0f2» додана офіційна підтримка «Visual Studio C#» для операційних систем «Windows» та «macOS». В минулих версіях використовувався проект «MonoDevelop-Unity». «Visual Studio» підтримує «.NET 4.6», що дозволить використовувати повний функціонал мови програмування «C#» версії 6.0 та більш нових версій. Також в «Unity» доступна технологія «Universal Render Pipeline». Ця технологія дозволяє за допомогою C# налаштовувати рендерінг, що в свою чергу дозволить мінімізувати вимоги до системи на якій буде працювати ігровий застосунок.

Розробники «Unity» спільно з компанією «Samsung» розробили сервіс «Adaptive Perfomance». Цей сервіс був створений для оптимізації роботи на мобільних пристроях застосунків розроблених з використанням ігрового рушія «Unity». З його допомогою ви можете керувати термічними показниками та змінювати якість графічної складової вашого застосунку. Використання цієї технології забезпечує покращення кадрової частоти під час гри.

Обираючи в якості рушія для розробки застосунку «Unity», ви отримуєте доступ до офіційного магазину ресурсів «Unity Asset Store». Доступ до магазину можна отримати не припиняючи роботу над проектом, на відповідній вкладці всередині ігрового рушія. Магазин включає в себе велику кількість спрайтів, моделей, плагінів для розробки 2D та 3D проектів. Пошук потрібних вам матеріалів здійснюється за допомогою фільтрів, є можливість сортування за ціною та рейтингом.

Для мобільних розробників доступний безкоштовний сервіс Unity Ads. З його допомогою можна додати рекламу та онлайн покупки для монетизації програмного продукту. Крім цього в «Unity» є підтримка графічного редактора «Adobe Photoshop» та редактора тривимірної графіки «Maya», ними часто користуються художники та дизайнери під час створення програмного забезпечення[7].

1.3 Операційні системи Android та iOS

Останні вісім років пальму першості серед мобільних операційних систем тримають Android та iOS. За даними сайту «Statcounter.com» станом на березень 2021-го року серед усіх операційних систем, включаючи комп'ютерні, Android найпопулярніша, вона займає 40,17%. iOS на третьому місці – 16,47%[3]. .В «Android» реалізована гнучка модель користувацького інтерфейсу. Користувач може налаштувати зовнішній вигляд своєї операційної системи під свої потреби та вподобання. Кастомізація в «iOS» не така обширна. Для того щоб опублікувати програму в «Google Play Market» потрібно зареєструвати обліковий запис, заплатити разовий внесок в розмірі 25$, завантажити свій застосунок. Через декілька годин користувачі зможуть встановлювати його на свої девайси. Алгоритм публікації в «App Store» дещо складніший, потрібно сплачувати щорічний внесок у розмірі 99$, після завантаження застосунку потрібно очікувати оцінку вашого проекту, зазвичай це два тижні. В «App Store» більш жорсткі критерії відбору публікованих програм, але й

платіжоспроможність користувачів девайсів «Apple» вища ніж у «Android». Для розробників, які не мають достатнього досвіду розробки «Android» буде більш пріоритетною платформою, тому що публікація застосунку простіша, а аудиторія користувачів більш обширна ніж у «iOS».

1.4 Середовище розробки Microsoft Visual Studio

Мова програмування «C#» створена компанією «Microsoft», тому найкращим варіантом вибору середовища розробки для «C#» буде «Microsoft Visual Studio». В випадках, коли потрібно використовувати технології «UWP» (Universal Windows Platform) і «WPF» (Windows Presentation Foundation) без «Visual Studio» не обійтись.

Дане середовище розробки має безкоштовну версію «Community Edition», для ознайомлення та розробки програмних продуктів для користувачів, які не мають можливості використовувати версії «Professional» «Enterprise». Також користувачам доступні розширення під певні задачі, для полегшення роботи з кодом. Прикладом такого розширення є «Visual Studio Tools for Unity». З його допомогою ви можете використовувати функції редагування та налагодження для написання скриптів для ігрових об’єктів «Unity». Оновлення в Visual Studio встановлюються в фоновому режимі і не заважають роботі. Якщо увійти в систему за допомогою облікового запису «Microsoft» або через «GitHub», то є можливість синхронізувати дані на двох або більше персональних комп’ютерах. Крім цього при реєстрації у системі вам доступний хмарний сервіс, використовуючи який ви можете завантажувати проекти дані вашого проекту в хмару. В Visual Studio є можливість використовувати утиліти, з їх допомогою вам доступний список служб, які виконуються, або пошук у вікні «Internet Explorer».

1.5 Тестування як інструмент для вивчення іноземної мови

В наш час вивчення іноземних мов є дуже актуальним. Знання провідних мов світу дає можливість отримати освіту у всесвітньо відомих вищих навчальних закладах США, Великої Британії, Канади. Для працевлаштування у великі ІТ-компанії обов'язковим є знання англійської мови на рівні вище середнього. Крім цього знання іноземної мови дозволяє використовувати в процесі навчання іншомовні джерела інформації, наприклад книги чи статті в журналах та в мережі, перегляд навчального матеріалу або прослуховування аудіокниги.

Тестування дуже зручний спосіб оцінити знання студентів або проводити регулярний контроль знань. За короткий проміжок часу можна отримати дані про рівень знань у тій чи іншій галузі. Крім цього, за допомогою тестів можна проводити навчання працівників на підприємствах. В умовах дистанційного навчання основною формою контролю знань є саме тестування. Тому тест може розглядатися як важливий компонент процесу освіти, від наповнення якого залежить рівень та якість процесу навчання, а також результат управління якістю педагогічних технологій. На сьогодні майже всі державні іспити в різних країнах мають у своєму складі тестову частину.

1.6 Огляд існуючих аналогів

Під час проектування був проведений огляд існуючих аналогів програмного забезпечення, що розробляється – «PlayLearn». Існуючі аналоги представлені в магазині «Google Play Market» та розповсюджуються на безкоштовній основі для платформи Android. Це ігри в жанрі 2D-платформер, які мають елемент головоломки.

Після проведеного огляду було складено порівняльну характеристику:

Таблиця1.1 – Порівняння з аналогами

Назва	Локалізація	Масштабованість інтерфейсу	Наявність онлайн магазину	Орієнтація екрану	Елемент головоломки	Елемент навчання
PlayLearn	Англійська, Українська	Так	Ні	Landscape	Так	Так
Kub-Puzzle Platformer	Англійська	Так	Так	Landscape	Так	Ні
Twinbotz	Англійська	Так	Ні	Portrait	Так	Ні
Tricky Castle	Англійська, Російська	Так	Так	Landscape	Так	Ні

Виходячи з даних таблиці до переваг застосунку «PlayLearn» можна віднести:

- українську мову локалізації інтерфейсу;
- наявність елементу навчання.

1.7 Висновки

На початку роботи над програмним продуктом, був проведений аналіз ринку мобільних пристроїв та застосунків для мобільних операційних систем. Найбільш популярною та гнучкою мобільною системою виявився «Android». Оглянуті та описані інструменти для розробки, виявлені їх плюси та мінуси. Також було обрано декілька аналогів програмного продукту та складено порівняльну таблицю, в якій показані переваги продукту перед його аналогами. Програмні продукти, які обирались для порівняння розповсюджуються на безкоштовній основі.

В результаті дослідження в якості рушія для розробки буде виступати «Unity», через його популярність, доступність та зручність в розробці для мобільних платформ. Під час аналізу рушія було виявлено найкращу мову програмування для роботи з ним. Через офіційну підтримку та можливість використання повного спектру функцій рушія розроблятись застосунок буде на мові програмування «C#». Середовище розробки «Microsoft Visual Studio» , тому що воно має низку переваг під час роботи на «C#» зокрема використання «UWP» та «WPF». Також це середовище має безкоштовну версію, якої достатньо для комфортної роботи з «Unity». Встановити середовище та мову програмування можна під час встановлення рушія.

За елемент навчання будуть відповідати тестові запитання з англійської мови на які потрібно потрібно давати відповідь перед початком кожного рівня. Англійська мова обрана через її популярність та через можливість отримання освіти в іноземних вищих навчальних закладах. Знання іноземної мови буде дуже корисним вмінням під час складання резюме для працевлаштування. Роботодавець високо цінує співробітників, які добре володіють англійською та можуть працювати з інформацією на іншомовних сайтах.

Результатом аналізу ринку стали:

- Складена порівняльна характеристика;
- Збір та вивчення даних щодо покращення програмного продукту.

Після аналітичної частини роботи будуть складені специфікації до системи програмного продукту з урахування недоліків та переваг аналогів.

2 ВИМОГИ ДО СИСТЕМИ

2.1 Види та рівні вимог програмного забезпечення

Вимоги до програмного забезпечення – це набір специфікацій, які повинні бути реалізовані в програмному продукті, що розробляється.

Прийнято розрізняти три рівні вимог до програмного забезпечення:

- Бізнес-вимоги;
- Користувацькі вимоги;
- Функціональні вимоги.

Бізнес-вимоги описують економічну складову проекту, спосіб отримання прибутку. Користувацькі вимоги визначають способи взаємодії програмного забезпечення з користувачем. Функціональні вимоги описують поведінку системи, основні її функції.

Окрім вищезазначених вимог існують ще і нефункціональні вимоги до програмного забезпечення. Вони являють собою опис того, як повинен працювати програмний продукт, вимоги до його безпеки, продуктивності, зовнішнього вигляду та обмеження, які описують можливі варіанти реалізації цих вимог.

Системні вимоги до продукту визначають високорівневі вимоги, які включають в себе велику кількість підсистем програмного забезпечення та обладнання. Користувач також може бути частиною системи, тому функції системи можуть стосуватися і користувачів.

Бізнес-правила – це політика компанії, стандарти якості, державні постанови. Бізнес-правила не відносяться до функцій системи, але вони впливають на обмеження системи, визначають правила, яких повинна дотримуватись система.

Атрибути якості являють собою додаткову характеристику нефункціональних вимог, які важливі для користувача. До таких вимог відносять легкість використання, ефективність, надійність.

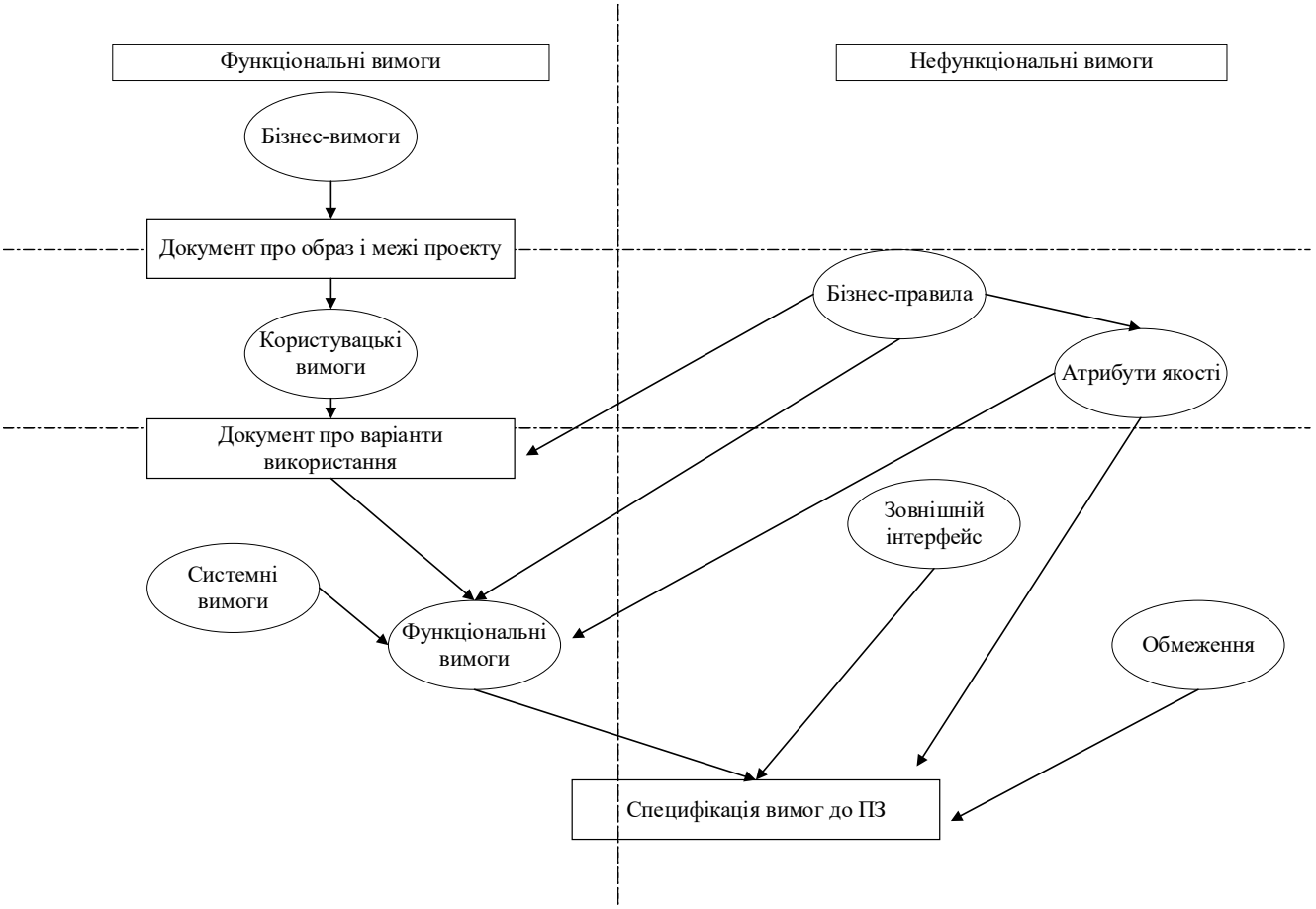


Рисунок 2.1 – Взаємозв’язки типів вимог до ПЗ

2.2 Історії користувача(User stories)

Представити список користувацьких вимог до системи можна за допомогою історій користувача (user stories). User story будується за певним шаблоном, спочатку вказується персонаж, який хоче отримати певний функціонал, потім описується, який функціонал хоче отримати персонаж і навіщо він хоче його отримати. Історії

користувача складаються від першої особи. Шаблон виглядає таким чином: «персонаж» хоче отримати «функціонал», для того щоб «причина».

Використовуючи даний шаблон були складені історії користувача для застосунку – «PlayLearn»:

- Як користувач, я хочу мати можливість змінювати мову локалізації, для комфортного користування застосунком;
- Як користувач, я хочу переглядати останню статистику рівнів, які я пройшов, для відстежування свого прогресу;
- Як користувач, я хочу бачити таймер на кожному рівні, для відстежування часу на проходження;
- Як користувач, я хочу мати можливість поставити гру на паузу, для перерви;
- Як користувач, я хочу здійснювати вільний вибір між пройденими рівнями, для повторного проходження;
- Як користувач, я хочу вимикати звук, для використання застосунку в людних місцях;
- Як користувач, я хочу мати кнопку виходу, для того щоб завершити використання застосунку.

За допомогою історій користувача ми можемо побачити, що користувач хоче отримати під час використання програмного продукту, необхідні функції та причини їх виникнення.

2.3 Варіанти використання (Use cases)

Варіанти використання – це послідовний опис взаємодії користувача та системи. Сценарій реалізації певної функції. Під час проектування програмного забезпечення за допомогою варіантів використання були представленні такі вимоги до програмного продукту – «PlayLearn»:

Після запуску додатку, в головному меню потрібна можливість зміни мови локалізації. В лівій верхній частині екрану представлені можливі мови локалізації для вибору. За замовчуванням встановлена англійська мова локалізації(табл. 2.1).

Таблиця 2.1 – Зміна мови локалізації

Use case 1.	
Діючі особи	Користувач, система.
Ціль	Зміна мови локалізації.
Сценарій: - Користувач запускає застосунок; - В головному меню натискає на кнопку відповідно обраній мові.	
Результат	Мова локалізації змінена.

Для перегляду статистики пройдених рівнів буде реалізоване відповідне меню. Перехід здійснюється в головному меню за допомогою кнопки «Levels» (табл. 2.2).

Таблиця 2.2 - Перегляд статистки

Use case 2.	
Діючі особи	Користувач, система.
Ціль	Перегляд статистики пройдених рівнів.

Таблиця 2.2 - Перегляд статистки

Сценарій: - Користувач запускає застосунок; - В головному меню натискає на кнопку, яка відкриває меню рівнів; - Статистика кожного рівня відображається у відповідному віконці для кожного пройденого рівня.	
Результат	Статистика переглянута.

Під час проходження рівнів, та в головному меню повинна бути реалізована можливість вимкнення звукових ефектів. На екрані паузи та в головному меню в верхній правій частині екрану буде знаходитись кнопка для керування звуком(табл. 2.3).

Таблиця 2.3 - Вимкнення звуку

Use case 3.	
Діючі особи	Користувач, система.
Ціль	Вимкнення звуку.
Примітка	Звук увімкнено за замовчуванням.
Сценарій: - Користувач запускає додаток; - Користувач вимикає звук за допомогою відповідної кнопки в головному меню або в меню паузи.	
Результат	Під час гри та після перезапуску застосунку звук буде вимкнено.

Для перезапуску поточного рівня під час використання застосунку потрібно відкрити вікно паузи за допомогою відповідної кнопки на ігровій сцені. Після цього натиснути на кнопку в лівій верхній частині вікна.

Таблиця 2.4 - Перезапуск поточного рівня

Use case 4.	
Діючі особи	Користувач, система.
Ціль	Перезапуск поточного рівня.
Сценарій: - Користувач під час проходження рівня за допомогою кнопки в правому верхньому кутку вмикає паузу; - У вікні паузи натискає на кнопку, яка відповідає за перезапуск поточного рівня.	
Результат	Перезапуск поточної сцени виконаний.

Для повторного проходження рівнів, потрібно в меню рівнів натиснути на обраний рівень(табл. 2.5).

Таблиця 2.5 - Вибір рівня

Use case 5.	
Діючі особи	Користувач, система.
Ціль	Вибір рівня.
Примітка	Вибір відбувається тільки між пройденими рівнями.
Сценарій: - Користувач проходить рівень; - Після проходження користувач повертається до головного меню;	

Таблиця 2.5 – Вибір рівня

- За допомогою кнопки «Levels» відкриває меню вибору рівня; - Обирає рівень.	
Результат	Рівень обрано.

На початку рівня, користувач повинен відповісти на запитання тесту, шляхом натискання на кнопку із обраним варіантом відповіді, у разі неправильної відповіді буде відтворений характерний звуковий ефект та максимальна оцінка за відповідь знизиться, якщо відповідь правильна, вікно з тестом закривається та супроводжується відповідним звуковим ефектом. На ігровій сцені з’являється кількість набраних балів та запускається відлік часу.

Таблиця 2.6 - Відповідь на запитання тесту

Use case 6.	
Діючі особи	Користувач, система.
Ціль	Відповідь на запитання тесту.
Успішний сценарій: - На початку рівня відкривається панель із тестом; - Користувач обирає правильний варіант відповіді; - Панель з тестом зникає, вгорі екрану з’являється кількість балів(максимум 100) та відлік часу.	
Результат	Відповідь обрана
Розширення:	
а	Обраний неправильний варіант. - Кнопка обраного варіанту змінює колір на червоний; - Оцінка знижується на 10

2.4 UML-діаграми варіантів використання та діяльності

Сценарії взаємодії користувача і системи, також, можна описувати за допомогою UML-діаграми варіантів використання. Основні компоненти цієї діаграми:

- Діюча особа (actor) – користувач системи;
- Варіант використання (use case) – сценарій взаємодії користувача з системою;
- Взаємозв'язки між діючою особою і варіантом використання (include, extend).

  1) Include це тип взаємозв'яку, який показує що саме використовує базовий варіант для виконання задачі і обов'язковим є виконання хоча б одного варіанту.

  2) Extend це тип взаємозв'язку, який розширює варіант використання додатковими функціями.

Використання цих функцій не є обов'язковим до виконання для продовження роботи з програмним забезпеченням.

Діаграма варіантів використання показує функції, які повинна виконувати система, а діаграма діяльності показує процес виконання функцій системи. Діаграма діяльності – це візуальне представлення графу, вершинами цього графу виступають певні дії, переходи між ними здійснюються по завершенню дії. Основні компоненти діаграми дільності:

- Скруглений прямокутник – дія;
- Ромб – рішення;
- Риски – початок або кінець паралельних процесів;
- Чорний кружок – початок діяльності;
- Обведений чорний кружок – кінець діяльності;
- Стрілки – порядок виконання функції.

В ході проектування програмного забезпечення за допомогою UML-діаграм варіантів використання були представлені основні функції системи, взаємодія між

ними та взаємодія користувача з системою. За допомогою UML-діаграм діяльності були описані процеси виконання функцій системи.

На діаграмі (рис.2.2), зображено головне меню застосунку. Під час взаємодії користувач може почати нову гру, перейти в меню вибору рівнів, змінити мову локалізації, керувати звуковими ефектами та закрити застосунок. Локалізація включає в себе переклад елементів інтерфейсу, таких як кнопки та текстові поля. В застосунку реалізована можливість англійської та української локалізації на вибір. Під час виходу система зберігає налаштування звуку, мову локалізації та кількість пройдених рівнів.

Рисунок 2.2 – Головне меню

На початку рівня користувач повинен відповісти на запитання тесту, якщо обраний правильний варіант, користувач отримує максимальний бал, який буде відображено вгорі екрану мобільного пристрою. Якщо обрано хибну відповідь, максимальний бал знижується на 10 і користувач обирає варіант відповіді повторно (рис. 2.3).

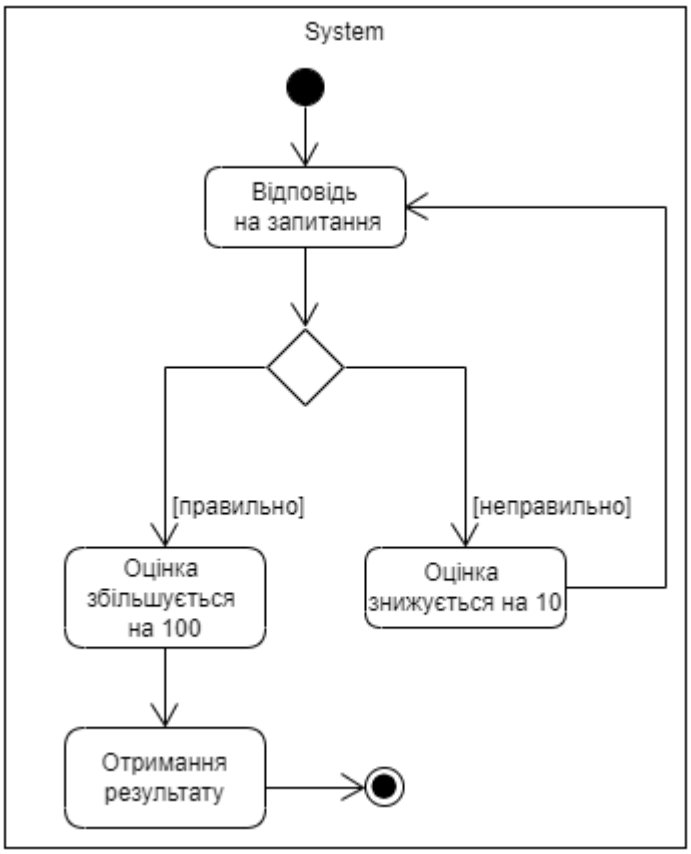


Рисунок 2.3 – Відповідь на запитання тесту

Під час проходження рівня користувач має можливість поставити застосунок на паузу. Після цього таймер зупиняється і відображається відповідне вікно. У вікні користувач може почати спочатку, перейти до головного меню, вимкнути або увімкнути звукові ефекти або продовжити проходження рівня. Після повернення до проходження рівня відлік часу продовжиться (рис. 2.4).

Рисунок 2.4 – Меню паузи

У випадку вірної відповіді на запитання починається процес проходження рівня, у разі програшу ігрова сцена перезапускається і користувач повертається до етапу відповіді на запитання. Якщо користувач проходить рівень, то відбувається збереження часу проходження рівня та кількості балів, які користувач набрав в процесі відповіді на запитання (рис. 2.5). Після збереження даних відображається меню виграшу, в якому користувач може перейти до наступного рівня, перезапустити рівень або повернутись до попереднього рівня.

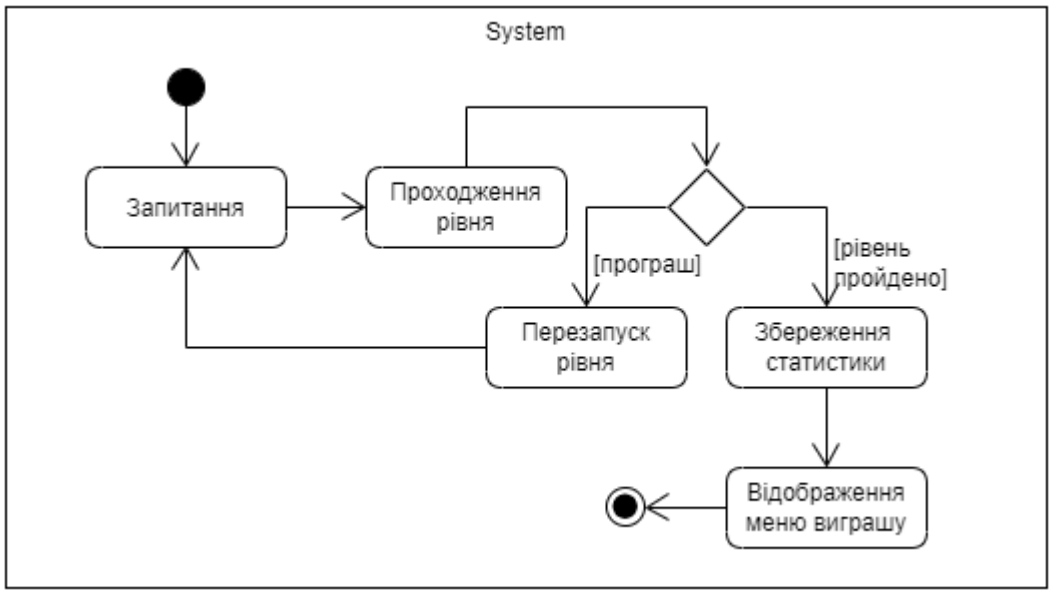


Рисунок 2.5 – Процес проходження рівня

Після проходження рівня час та кількість балів, які набрав гравець будуть збережені та показані на панелі виграшу. Крім цього збережені дані будуть відображатися при виборі рівня.

Щоб показати взаємодію системи та користувача, запити, які надходять до системи та потік даних доцільно зобразити це за допомогою сценарію прецедента (рис. 2.6). На цій діаграмі можна побачити запити які надходять до системи, та реакцію системи на них. Які методи викликаються. Показано де починається процес та де завершується.

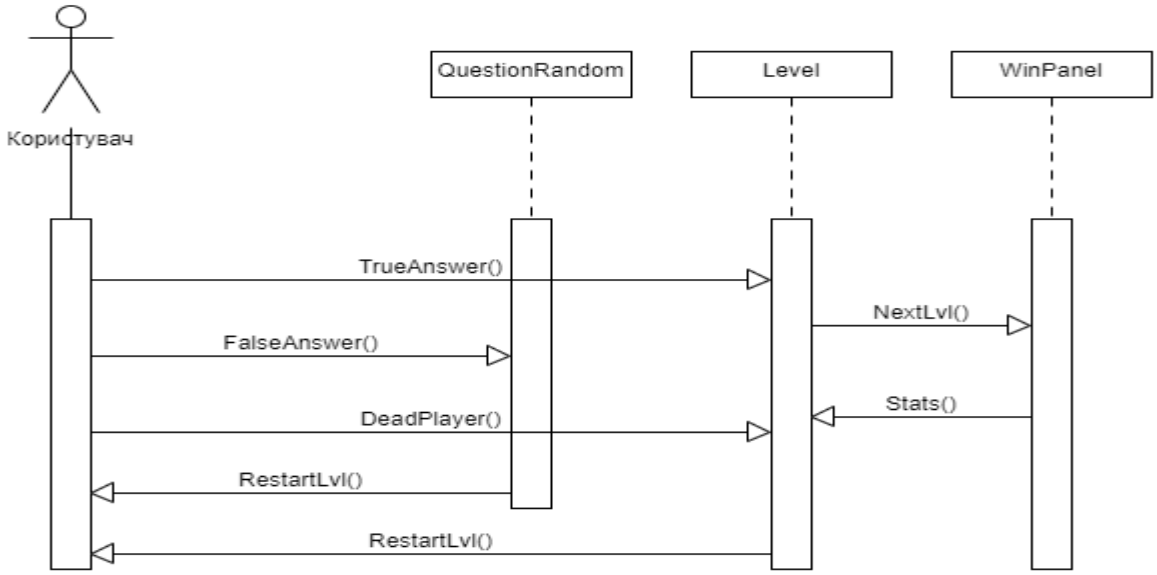


Рисунок 2.6 – Сценарій прецедента «Проходження рівня»

Для відображення статистики усіх пройдених рівнів, користувачу необхідно перейти до меню вибору рівнів. Зробити це можна за допомогою відповідної кнопки в головному меню. В меню вибору рівнів наведений список всіх рівнів та статистика пройдених користувачем рівнів. Вибір здійснюється між пройденими рівнями, якщо таких немає, то для проходження доступний лише перший рівень (рис. 2.7).

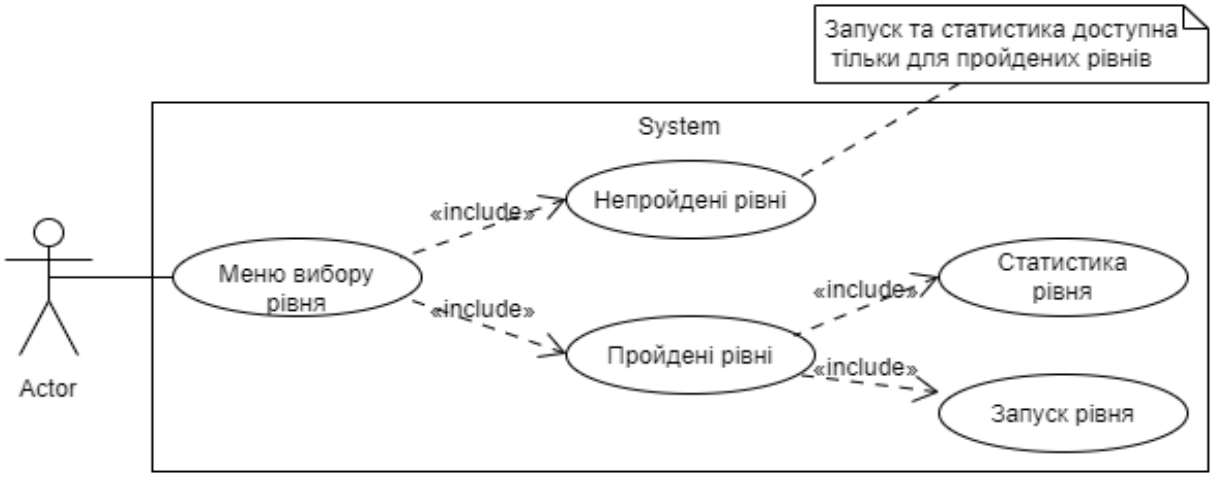


Рисунок 2.7 – Меню вибору рівнів

Для кожного рівня існує масив запитань та відповідей. Після перезапуску рівня запитання та варіанти відповіді змінюються. На кожне запитання є тільки одна правильна відповідь (рис. 2.8).

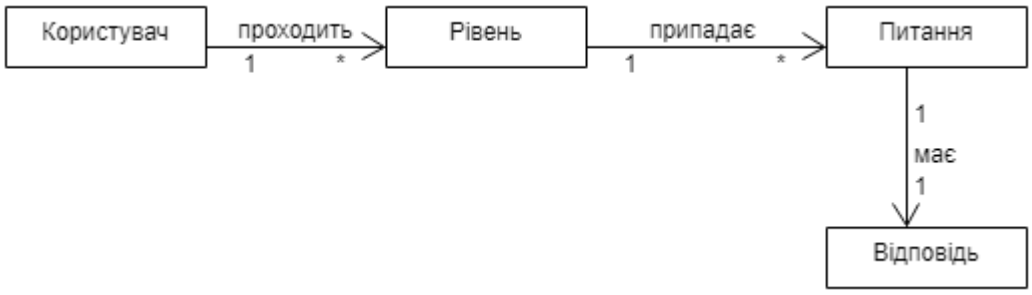


Рисунок 2.8 – Діаграма предметної галузі

2.5 Висновки

На етапі проектування були визначені основні вимоги до програмного забезпечення. Були складені історії користувача, які показують, які можливості потрібні користувачеві під час використання застосунку. Також були написані сценарії використання, для розуміння взаємодії користувача із системою. Для графічного представлення вимог використовувались UML-діаграми варіантів використання та діяльності.

Під час виконання цього етапу розробки були покращені теоретичні та практичні знання з аналізу вимог до програмного забезпечення.

3 РОЗРОБКА ТА ТЕСТУВАННЯ ІГРОВОГО ЗАСТОСУНКУ

3.1 Початок роботи. Встановлення середовища розробки та завантаження необхідних матеріалів

Для початку роботи необхідно встановити ігровий рушій Unity, зробити це можна на офіційному сайті розробника[10]. Під час встановлення ігрового рушія є можливість встановити Microsoft Visual Studio 2019 для розробки з використанням мови програмування C#. Після встановлення потрібно відкрити Unity та створити новий проект (рис 3.1).

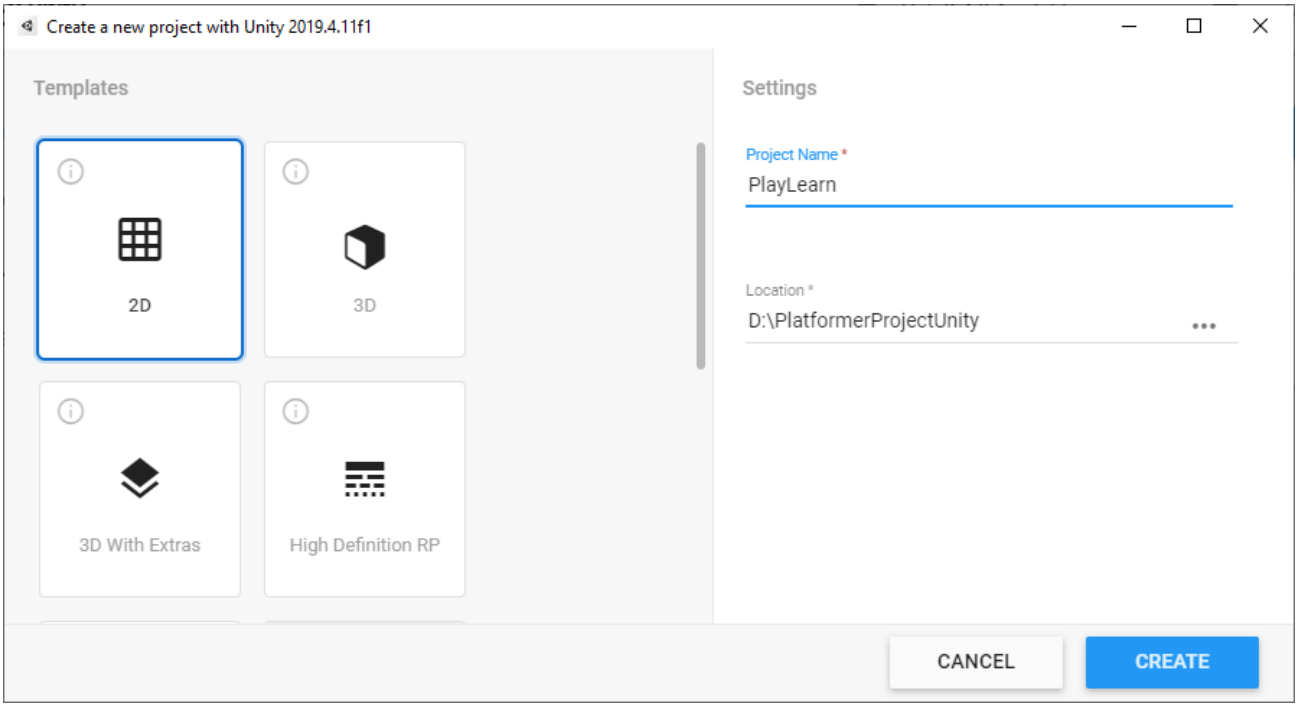


Рисунок 3.1 – Створення нового проекту

Unity на вибір є 4 шаблони, кожен шаблон це певна стартова конфігурація параметрів проекту, вибір шаблону полегшує підготовку до розробки. Для свого

проекту я обрав шаблон 2D. Далі потрібно вказати місце зберігання файлів проекту та його назву. Проект матиме назву «PlayLearn».

Для розробки ігрового застосунку в 2D необхідні спрайти, фізичні матеріали та інші ресурси. Завантажити їх можна використовуючи інтегрований магазин Unity – Unity Asset Store (рис. 3.2).

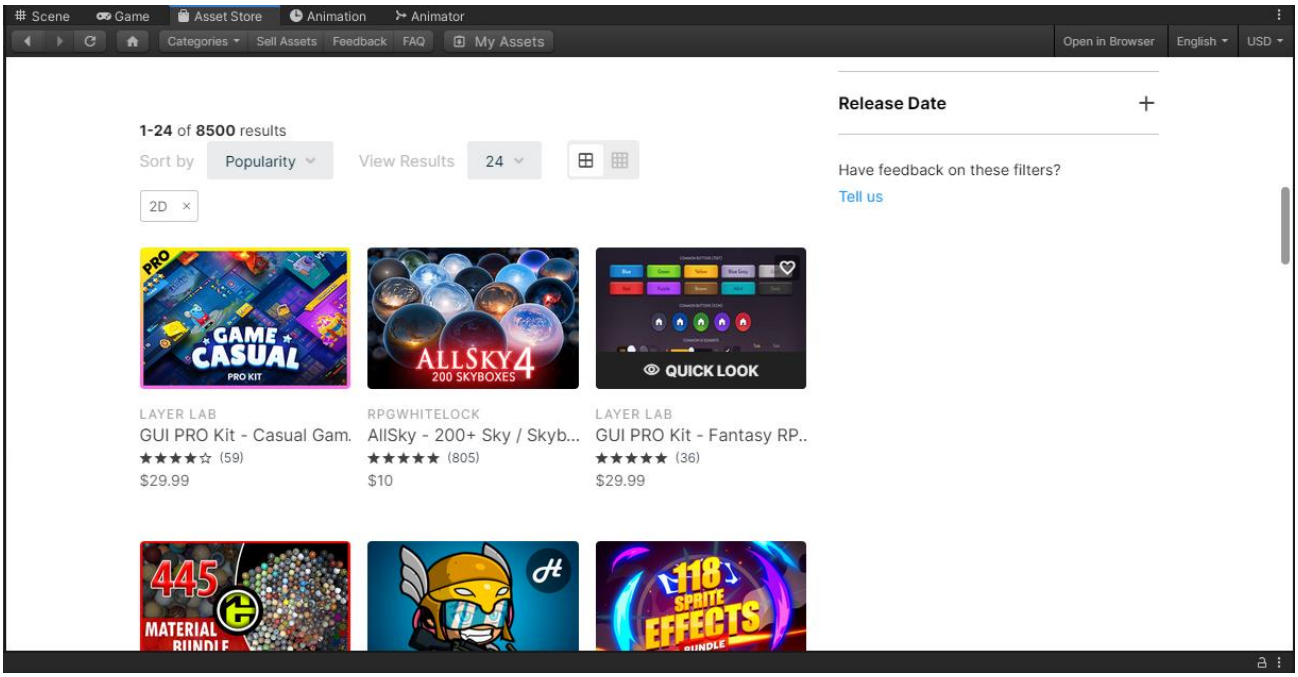


Рисунок 3.2 Інтегрований магазин ресурсів Unity

Для розробки будуть використані безкоштовні пакети ресурсів, які включають в себе:

- Спрайти головного героя;
- Елементи заднього фону;
- Звукові ефекти;
- UI елементи.

Після завантаження пакетів ресурсів імпортуємо їх в Unity.

3.2 Дизайн рівнів

Кожен рівень буде мати вигляд ігрової сцени, на якій будуть розміщені ігрові об’єкти такі як платформи, задній фон, пастки. Персонаж, яким керує гравець знаходиться на початку рівня. Для завершення рівня персонаж повинен перетнути тригер, який знаходиться у відповідному місці. Рівень проходиться зліва направо. При потраплянні в пастку рівень починається спочатку (рис. 3.3).

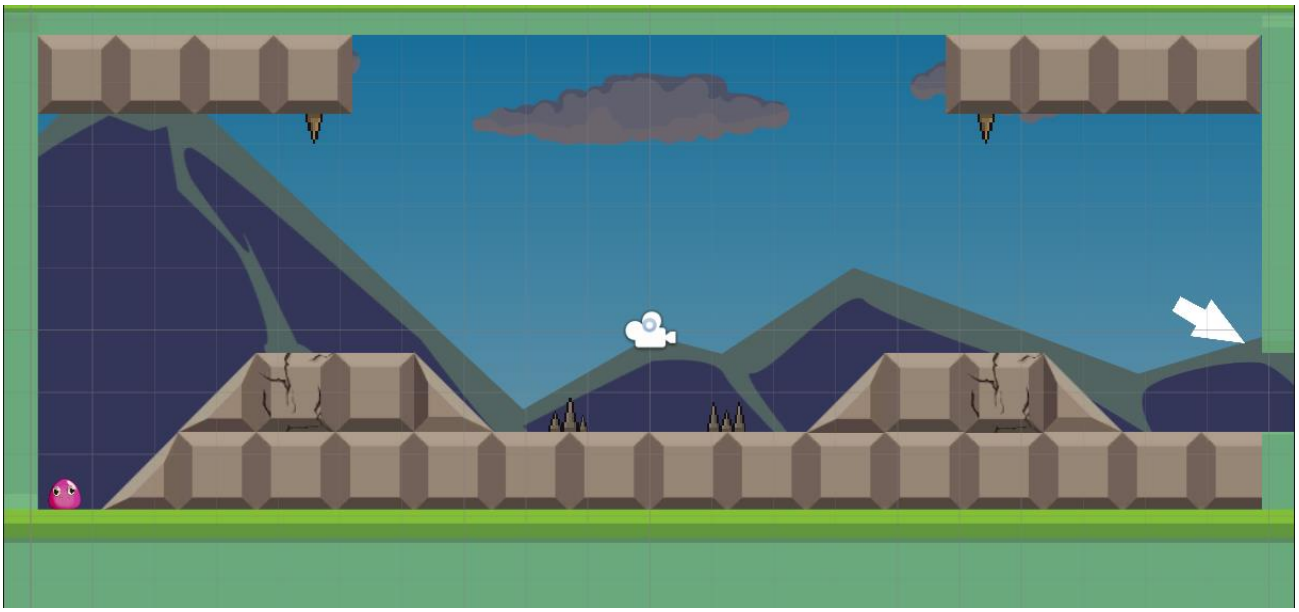


Рисунок 3.3 – Ігрова сцена

В якості пасток будуть виступати шипи. Контакт персонажа з шипами буде зарахований як програш. Всього буде доступно 15 рівнів для проходження. Платформи будуть поділятися на чотири види, а саме:

- Статичні;
- Рухомі;
- Зникаючі;
- Стрибкові.

Графічно зобразити архітектуру платформ можна за допомогою UML-діаграми архітектури:

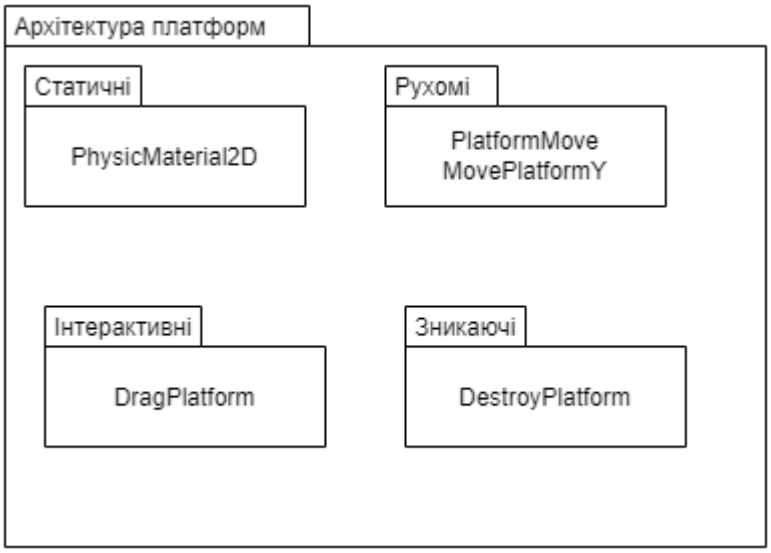


Рисунок 3.4 – Архітектура платформ

Користувач, який керує персонажем, має можливість рухатися праворуч, ліворуч та робити стрибок. На деяких рівнях з певними ігровими об’єктами можна взаємодіяти не через елементи користувацького інтерфейсу а напряму, наприклад перетягувати їх.

3.3 Написання скриптів

Для створення скрипта в Unity потрібно на вкладці «Project» натиснути праву кнопку миші та обрати «Create C# Script». Після цього відкрити його за допомогою середовища розробки Microsoft Visual Studio 2019. Для реалізації керування персонажем розробимо два скрипти. В першому надамо список команд а в другий будемо передавати ці команди та виконувати їх.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
   public enum ActionType
   {
      None = 0,
      MoveRight = 1,
      MoveLeft = 2,
      Jump = 3,
      Die = 4,
  }
```

Список команд виглядає таким чином, команда *«None»* відповідає за завершення дії, тобто якщо користувач натиснув та утримує кнопку яка відповідає за рух праворуч, то відбувається команда *«MoveRight»*, а коли користувач відпустив кнопку спрацьовує команда *«None»* і персонаж припиняє рух. Керування персонажем здійснюється за допомогою кнопок, тому потрібно передати значення команд на ці кнопки, так як через назву команди це зробити не можна, потрібно написати метод який буде приймати числове значення цих команд. Метод напишемо в другому скрипті.

```
public void OnDoAction(int actionld)
   {
      OnDoAction((ActionType)actionld);
  }
```

Для реалізації руху по горизонталі створимо дві змінні типу *«bool»*, які будуть визначати рух персонажа, та напрямок. Тепер напишемо метод який приймає значення *«ActionType»* та присвоює значення змінним типу *«bool»*.

```
public void OnDoAction(ActionType action)
```

```
  {
     if (action == ActionType.MoveRight)
     {
        this.onMoving = this.directionRight = true;
     }
     else if (action == ActionType.MoveLeft)
     {
        this.directionRight = false;
        this.onMoving = true;
     }
     else if (action == ActionType.None)
     {
        this.onMoving = false;
     }
  }
```

Коли метод отримує команду руху праворуч, змінна руху на напрямку праворуч стають *true*, якщо команда руху ліворуч, то напрямок праворуч *false*, а змінна руху *true*. При отриманні команди *«None»* рух припиняється. Аналогічно передається команда для стрибка, але потрібно написати метод, який перевіряє чи знаходиться персонаж на землі. Якщо цього не зробити, то при натисканні кнопки стрибка декілька разів підряд, персонаж, яким керує користувач буде робити безперевні стрибки.

За допомогою методу ми можемо перевіряти де знаходиться персонаж, якщо він буде знаходитись у повітрі, то стрибок буде неможливий.

```
private bool CheckGround()
 {
      Bounds bounds = this.collider.bounds;
      Vector2 point = bounds.center;
      Vector2 size = bounds.size;
      bool result = Physics2D.OverlapBox(point, size, 0f,layer);
      return result;
 }
if (CheckGround())
  {
      this.onMakeJump = true;
}
if (this.onMakeJump)
   {
       float xVelocity = this.body.velocity.x;
       float yVelocity = this.jumpForce;
```

```
        Vector2 force = new Vector2(xVelocity, yVelocity);
        this.body.AddForce(force, ForceMode2D.Impulse);
        this.onMakeJump = false;
}
```

Тобто, якщо персонаж знаходиться на об’єкті, який має шар з певною назвою, яку ми вкажемо в інспекторі, то змінна *«OnMakeJump»* стане «*true»* і стрибок може бути виконаний, доки персонаж знаходиться в повітрі змінна матиме значення *«false»*.

Тепер потрібно реалізувати механіку програшу при контакті з шипами. Для цього створимо окремий скрипт «DeadPlayer».

```
  private void OnTriggerEnter2D(Collider2D collision)
  {
     if (collision.gameObject.tag == "Player")
     {
        Destroy(collision.gameObject);
     }
  }
```

Метод *«OnTriggerEnter2D»* викликається при контакті об’єкта з будь-яким іншим об’єктом в якого є компонент *«Collider2D».* При контакті об’єкта, який має тег *«Player»* відбудеться знищення цього об’єкта.

Окрім звичайних статичних платформ потрібно зробити рухомі та зникаючі.

```
void Update()
  {
     if (transform.position.y > rangeMoveUp)
     {
        moveUp = false;
     }
     else if (transform.position.y < rangeMoveDown)
     {
        moveUp = true;
     }
     if (moveUp)
     {
        transform.position = new Vector3(transform.position.x, transform.position.y + speed *
Time.deltaTime,transform.position.z);
     }
     else
     {
```

```
        transform.position = new Vector3(transform.position.x, transform.position.y - speed *
Time.deltaTime,transform.position.z);
    }
}
```

Метод *«Update»* викликається в реальному часі. Кількість його викликів дорівнює кількості кадрів за секунду. Метод буде перевіряти значення змінних, якщо значення буде більше ніж ми вказали, вектор руху змінюється.

Метод для реалізації платформ, які зникають буде спрацьовувати при контакті персонажа з нею.

```
private void OnCollisionEnter2D(Collision2D collision)
  {
    if (collision.gameObject.tag == "Player")
      Destroy(gameObject, timeToDestroy);
 }
```

Під час проходження рівня потрібно відстежувати час проходження і відображати його в реальному часі на екрані. Для цього потрібно створити об’єкт «Canvas», на ньому ми можемо створювати елементи користувацького інтерфейсу. Потрібно створити два поля типу *«Text»* і написати скрипт який буде відповідати за таймер та за кількість балів за рівень. Елементи керування для персонажа також розміщуються на об’єкті *«Canvas».* Якщо потрібно зробити опрацювання натискання на UI-елемент, необхідно на ігрову сцену додати *«Event System»,* цей об’єкт дозволить нам опрацювати події, такі як натискання. Для реалізації руху персонажа по горизонталі та для стрибка на відповідні кнопки додамо дві події – *«OnPointerDown»* та *«OnPointerExit».* Перша подія спрацьовує при натисканні на елемент, друга подія після завершення натискання.

```
void Start()
  {
    timerText.text = timeStart.ToString("F2");
```

```
    }
    void Update()
    {
        timeStart += Time.deltaTime;
        timerText.text = timeStart.ToString("F2");
}
```

Метод *«Start»* спрацьовує одразу після запуску сцени. На початку рівня ми запишемо в текстове поле значення таймеру. А в методі *«Update»* будемо збільшувати цей таймер та записувати його після кожного збільшення. *«ToString(“F2*”)» означає, що значення таймеру конвертується з числового типу в строковий та записується з двома символами після коми.

```
public void TrueAnswer()
    {
        score += 100;
        scoreText.text += " " + score;
    }
    public void FalseAnswer()
    {
        score -= 10;
    }
```

Якщо користувач відповів правильно на запитання, йому нараховується 100 балів, кожна неправильна відповідь зменшує оцінку на 10 балів. Графічно структуру класу «Stats» можна зобразити використовуючи діаграму класів (рис.3.5). На ній показані поля та методи класу, а також модифікатори доступу до них.

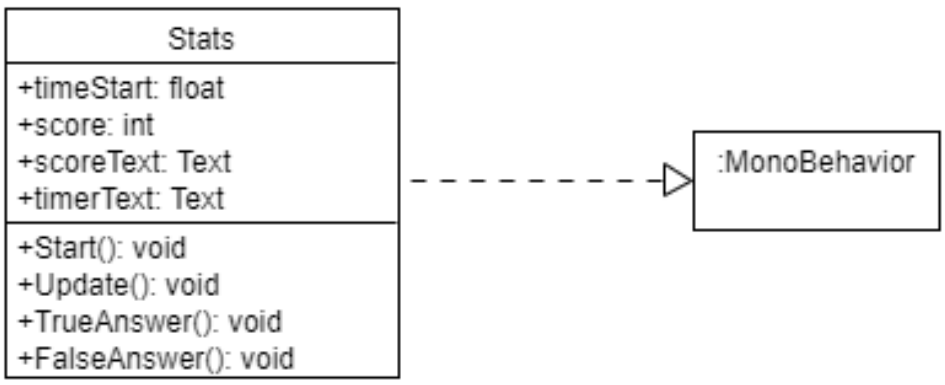


Рисунок 3.5 – Діаграма класу «Stats»

Щоб завершити рівень користувачеві необхідно дістатись до кінця сцени, потрібно створити об’єкт, додати компонент *«Collider2D»* та в інспекторі обрати функцію *«isTrigger»*. Потрібно написати скрипт, який буде зберігати пройдені рівні, щоб користувач міг обрати цей рівень для повторного проходження. Також потрібно статистику в меню вибору рівнів. Спочатку створимо меню вибору рівнів.

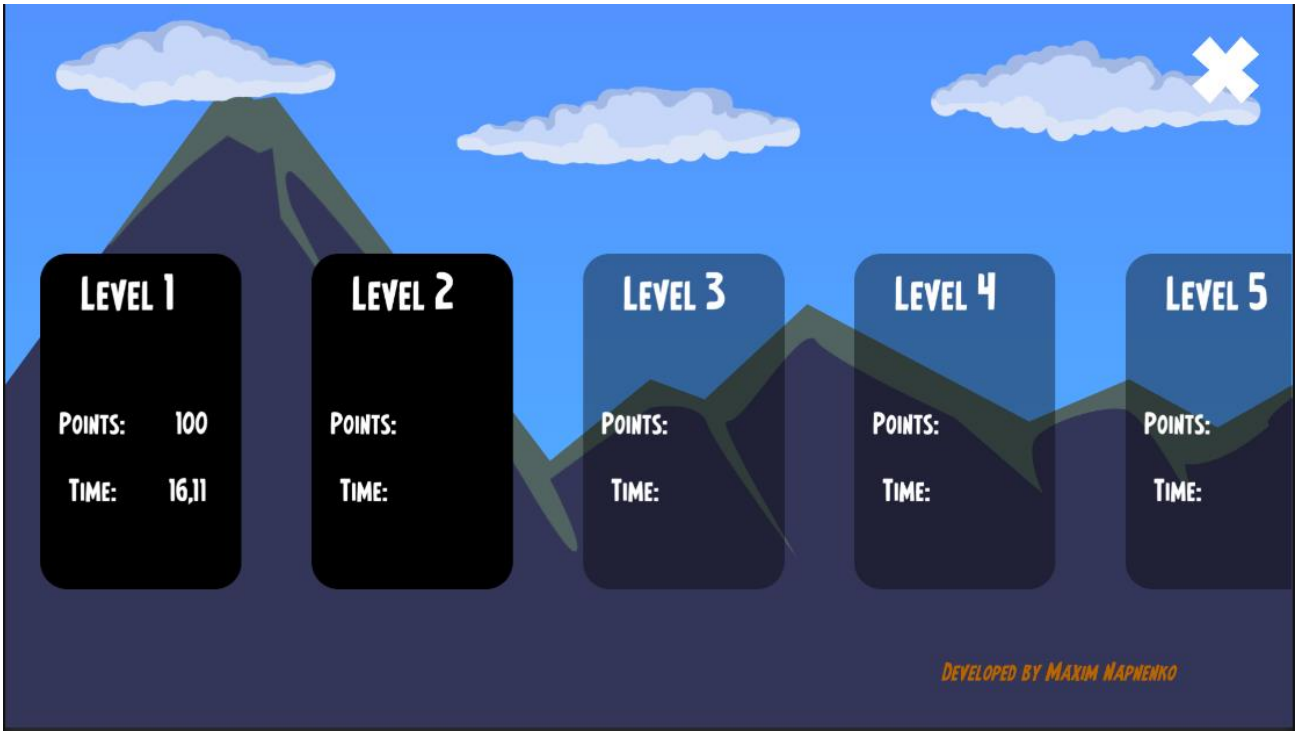


Рисунок 3.6 – Список рівнів

Кожен блок це кнопка, при натисканні відбувається запуск відповідного рівня. Створимо масив кнопок та змінну якій ми передамо значення пройдених рівнів. Запустимо цикл, який деактивує всі кнопки в масиві, а потім ще один цикл, який зробить активними кнопки, які відповідають за рівні які ми пройшли (рис.3.6).

```
private void Start()
   {
```

```
        levelUnlock = PlayerPrefs.GetInt("levels",1);
        for (int i = 0; i < buttons.Length; i++)
            buttons[i].interactable = false;
        for (int i = 0; i < levelUnlock; i++)
        {
            buttons[i].interactable = true;

        }
}
```

Тепер напишемо методи для запуску рівня і для початку нової гри. Щоб запустити рівень потрібно написати метод, який ми будемо викликати при натисканні на кнопку рівня та передамо в нього індекс рівня за який відповідає ця кнопка. А метод для початку нової гри видалить всі збережені дані та зробить доступним для проходження тільки перший рівень.

```
    public void loadLevel(int levelIndex)
    {
        SceneManager.LoadScene(levelIndex);
    }
    public void NewGame()
    {
        PlayerPrefs.DeleteAll();
        PlayerPrefs.SetInt("levels", 1);
        PlayerPrefs.Save();
    }
```

Тепер перейдемо до ігрової сцени і для об’єкту який буде зберігати пройдені рівні напишемо наступний код.

```
private void Start()
    {
        scene = SceneManager.GetActiveScene();
    }
    public void UnlockLevel()
    {
        int currentLevel = SceneManager.GetActiveScene().buildIndex;
        PlayerPrefs.SetInt("lvlnum", currentLevel - 1);
        if (currentLevel >= PlayerPrefs.GetInt("levels"))
```

```
        PlayerPrefs.SetInt("levels", currentLevel + 1);
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "Player")
        {
            UnlockLevel();
             }
```

Таким чином кожен пройдений рівень буде збільшувати змінну, яка знаходиться під ключем *«levels»* і в меню вибору рівнів буде збільшуватись кількість доступних рівнів.

На об’єкт, який відповідає за завершення рівня додамо скрипт який зберігає статистику рівня. Метод *«Savelvl»,* буде викликаний при перетинанні тригеру.

```
void Savelvl1()
    {
        PlayerPrefs.SetString("totalTime1", timer.text);
        PlayerPrefs.SetString("totalScore1", score.text);
        PlayerPrefs.Save();
    }
```

На кнопки в меню вибору рівнів додамо скрипт, який виводить збережену статистику.

```
public class Writelvl1 : MonoBehaviour
{
    public Text timerOnMenu1;
    public Text scoreOnMenu1;
    void Start()
    {
        timerOnMenu1.text = PlayerPrefs.GetString("totalTime1");
        scoreOnMenu1.text = PlayerPrefs.GetString("totalScore1");
    }
        }
```

3.4 Звукові ефекти

За відтворення звукових ефектів відповідає компонент Audio Source. Тому на об’єкти, які при взаємодії будуть відтворювати звук потрібно додати даний

компонент. Звук буде програватись коли ми натискаємо на елементи користувацького інтерфейсу, а також при відповіді на запитання.

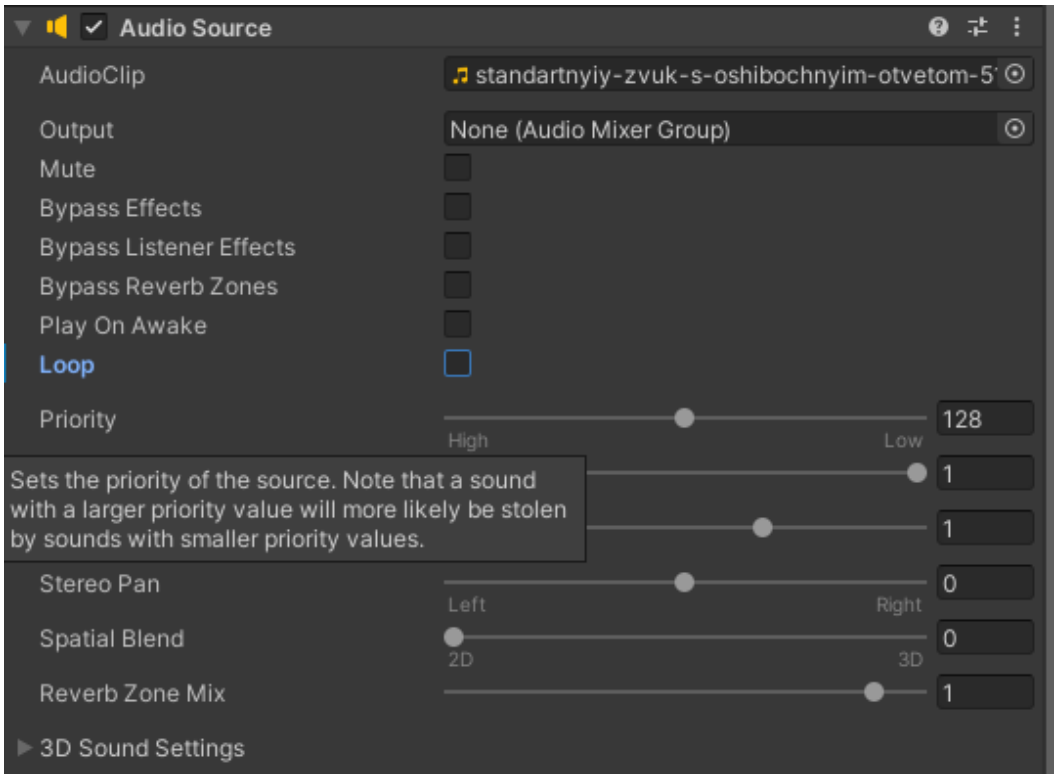


Рисунок 3.7 – Компонент «Audio Source»

Сюди ми додамо звуковий файл, який буде відтворюватися. Крім цього ми можемо активувати функцію «Play On Awake», що відтворює звук одразу після запуску ігрової сцени. Дану функцію потрібно активувати на візуальних ефектах які будуть створюватись за певних умов. Наприклад, якщо персонаж помирає на сцені створюється візуальний ефект вибуху, на якому буде компонент «Audio Source» з увімкненою функцією «Play On Awake». Тобто одразу після створення об’єкту буде програватись звук вибуху. Тепер потрібно написати скрипт, який буде відтворювати звук:

```
public class SoundEffectsUI : MonoBehaviour
{
    public AudioSource aSource;
```

```
void Start()
{
   aSource = GetComponent<AudioSource>();
}

public void StartPlaying()
{
   aSource.Play();
}
      }
```

За відтворення звукових ефектів відповідає рядок «*aSource.Play();*». Під час використання застосунку користувачеві потрібна можливість керування звуком. Тому в головному меню та в меню паузи потрібно додати елемент користувацького інтерфейсу та написати скрипт, який буде відповідати за увімкнення та вимкнення звукових ефектів. Графічно структуру класу «SoundEffectsUI» зображено на діаграмі (рис. 3.8).

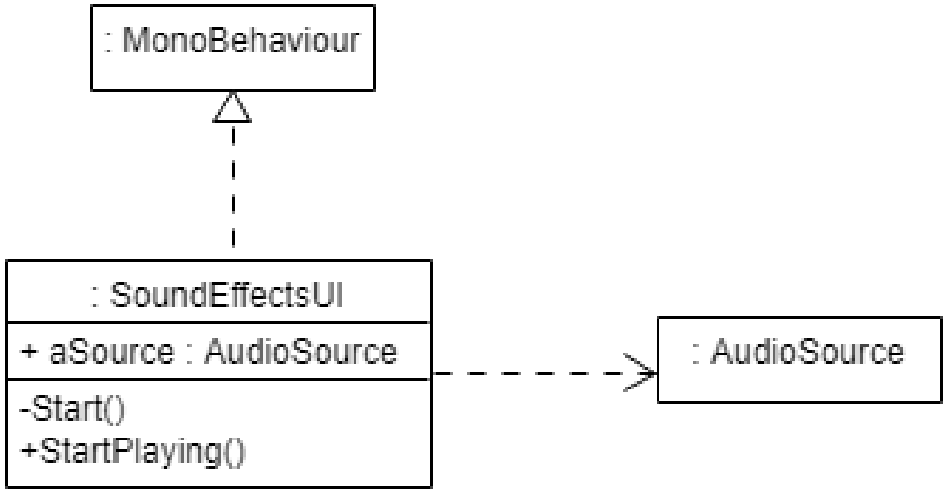


Рисунок 3.8 – Клас «SoundEffectsUI»

void Start()

```
  {
    checkImg = PlayerPrefs.GetInt("check");
    AudioListener.volume = PlayerPrefs.GetInt("sound");
    soundImg = GetComponent<Image>();
    if(checkImg==1)
      soundImg.sprite = soundOff;
    else if(checkImg==0)
      soundImg.sprite = soundOn;
 }
```

Спочатку ми беремо значення змінної «check» та змінної «sound». Перша змінна відповідає за спрайт кнопки, який буде на початку гри. Друга змінна відповідає за звук. Тобто, якщо ми вимкнемо звук та закриємо застосунок, а потім знову запустимо його, звукові ефекти залишаться вимкнутими та буде відображатись відповідний спрайт.

```
public void SwapImage()
  {
    if (soundImg.sprite == soundOn)
    {
      soundImg.sprite = soundOff;
      AudioListener.volume = 0;
      checkImg = 1;
      checkSnd = 0;
      PlayerPrefs.SetInt("check", checkImg);
      PlayerPrefs.SetInt("sound", checkSnd);
      return;
    }
    if (soundImg.sprite == soundOff)
    {
      soundImg.sprite = soundOn;
```

```
        AudioListener.volume = 1;
        checkImg = 0;
        checkSnd = 1;
        PlayerPrefs.SetInt("check", checkImg);
        PlayerPrefs.SetInt("sound", checkSnd);
        return;
    }
}
```

Метод «SwapImage()» буде спрацьовувати, коли ми натиснемо на кнопку керування звуком. Якщо звук увімкнено, то при натисканні він вимкнеться та зміниться спрайт кнопки, дані збережуться.

Результат роботи продемонстрований на діаграмі артефактів (рис. 3.9).

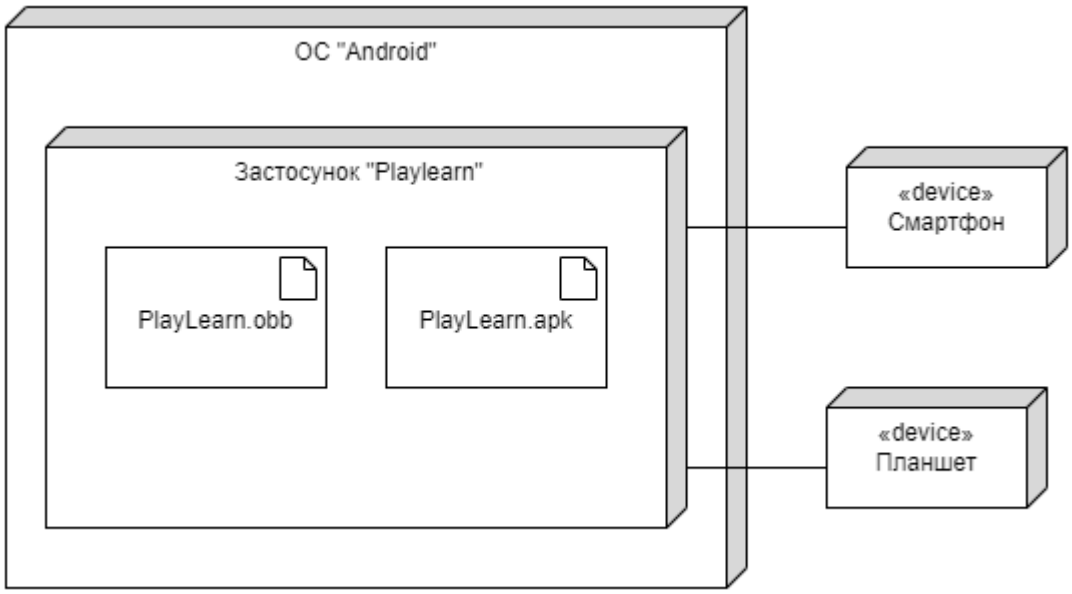


Рисунок 3.9 – Діаграма артефактів

3.5 Тестування застосунку

Тестування – це невід’ємна частина процесу розробки програмного забезпечення. Тестування потрібне для контролю якості ПЗ. З його допомогою можна перевірити ступінь відповідності продукту вимогам замовника. Для перевірки вимог потрібно описати тест-кейси. Застосунок буде перевірений на відповідність вимогам, які представленні в дипломній роботі.

Таблиця 3.1 - Тест-кейс 1

Номер	1
Назва	Зміна локалізації.
Кроки	1. Запустити застосунок. 2. У верхньому лівому кутку обрати мову локалізації. 3. Натиснути відповідну кнопку.
Результат	Мову локалізації змінено.

Таблиця 3.2 - Тест-кейс 2

Номер	2
Назва	Перегляд статистики пройдених рівнів.
Кроки	1. Завершити проходження рівня. 2. Перейти до головного меню. 3. Перейти до вибору рівня.
Результат	Статистика пройденого рівня відображається у відповідному блоці.

Таблиця 3.3 - Тест-кейс 3

Номер	3
Назва	Відображення таймера під час проходження рівня.
Кроки	1. Запустити будь-який доступний рівень.
Результат	У верхній частині екрану відображається таймер.

Таблиця 3.4 - Тест-кейс 4

Номер	4
Назва	Зупинка гри.
Кроки	1. Почати гру. 2. У верхньому правому кутку натиснути на кнопку паузи.
Результат	Таймер та всі рухомі об’єкти зупиняються.

Таблиця 3.5 - Тест-кейс 5

Номер	5
Назва	Вільний вибір між пройденими рівнями
Кроки	1. Пройти декілька рівнів. 2. Перейти до обрання рівня. 3. Обрати будь-який, раніше пройдений, рівень.

Таблиця 3.5 - Тест-кейс 5

Результат	Рівні запускаються.

Таблиця 3.6 - Тест-кейс 6

Номер	6
Назва	Керування звуком
Кроки	1. Запустити застосунок. 2. Праворуч вгорі натиснути на кнопку керування звуковими ефектами.
Результат	Звук вимкнено. При повторному натисканні відбудеться увімкнення звуку.

Таблиця 3.7 - Тест-кейс 7

Номер	7
Назва	Завершення роботи програми.
Кроки	1. Перейти до головного меню. 2. Натиснути кнопку «Вихід».
Результат	Завершення роботи програми.

3.6 Висновки

Розробка ПЗ проводилась із використанням інструментів, які були вибрані для роботи на початку розробки проекту. Спочатку були та завантажені пакети ресурсів, потім написані скріпти. Періодично, застосунок встановлювався на смартфон, для перевірки працездатності. Після завершення розробки, були проведені основні налаштування. Для цього використовувалась вкладка «Player Settings». Орієнтація екрана обрана «Landscape Right». Виконано налаштування ярлика гри та стартового екрану. Також було створено логотип за допомогою програми «Paint3D», який з’являється одразу після запуску програми.

Тестування проводилось для перевірки відповідності програмного продукту вимогам які були складені на етапі проектування. Звіт перевірки реалізований шляхом складання тест-кейсів. Тест кейси представлені таблицями.

За результатами проведеного тестування можна зробити висновок, що програмний продукт, який був розроблений в процесі виконання дипломного проекту відповідає всім вимогам, які були представлені. Всі поставлені на початку розробки задачі виконані, недоліки аналогів враховані. Програмні та технічні проблеми в процесі розробки не виникали. Критичних помилок, які порушують роботу програми або перешкоджають її роботі не виявлено.

4 ВИСНОВКИ

На сьогоднішній день мобільні пристрої є невід'ємною частиною життя сучасної людини. Кожного дня ми використовуємо смартфони, планшети, ноутбуки, смарт-годинники та інші девайси для виконання найрізноманітніших завдань. Однією з найпопулярніших категорій використання мобільних пристроїв є розваги. Популярність ігрових застосунків саме на смартфонах зумовлена тим, що доступ до своєї улюбленої гри можна отримати в будь-якому місці та в будь-який час. Тому розробка програмного забезпеченя під мобільні платформи є дуже актуальною та прибутковою.

Через те що ігри забирають багато часу в школярів та студентів, виникає проблема в навчанні. Учні не можуть сконцентруватися на вивченні матеріалів в школах та ВНЗ. Для вирішення цієї проблеми потрібно створити застосунок, який покращить процес навчання.

Результатом дипломної роботи є ігровий застосунок для мобільної операційної системи «Android» в жанрі 2D-платформер. Для навчання використовуються тестові запитання з англійської мови, на які користувач буде давати відповідь на початку кожного рівня.

Перед початком розробки були досліджені популярні аналоги програмного продукту. Були виявлені основні переваги та недоліки. На основі проведеного дослідження складено таблицю в якій порівнюються знайдені рішення та продукт, який буде розроблятись (табл. 1.1).

Під час вибору інструментів для розробки були оглянуті напопулярніші рішення, серед яких обрано найбільш зручні та підходящі для даного проекту.

В роботі проведено моделювання та проектування інформаційної системи:

- змодельовано модель предметної галузі (діаграма предметної галузі);
- змодельовано модель прецедентів (набір прецедентів та діаграми варіантів використання);

- проаналізовані нефункціональні вимоги в додатковій специфікації;
- змодельовано модель проектування (діаграми діяльності, послідовності та класів);
- відпрацьовано архітектуру програмного забезпечення, що дозволяє нарощувати нову функціональність та вносити нові дані. Тому вимоги до проекту можуть змінюватись.

В дипломній роботі виконана реалізація інформаційної системи:

- реалізовано програмне забезпечення;
- створено робочий проект, який є спроможний до використання в інших пов’язаних технологіях.

Таким чином за період розробки було покращено теоретичні та практичні знання щодо всього життєвого циклу програмного забезпечення. Проаналізовано ринок та зроблено висновки щодо актуальності розробки. Досліджена актуальність теми дипломної роботи та її практична значущість. Створено програмний продукт, який поністю відповідає вимогам, які були представлені. Готовий продукт може покращуватись та оновлюватись в процесі підтримки програмного забезпечення.

ПЕРЕЛІК ПОСИЛАНЬ

1. C# development with Visual Studio [Електронний ресурс] // Microsoft – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/?view=vs-2019.
2. Digital2020: Global Digital Overwiew [Електронний ресурс] // DataReportal. – 2020. – Режим доступу до ресурсу: https://datareportal.com/reports/digital-2020-global-digital-overview.
3. Operating System Market Share Worldwide [Електронний ресурс] // © StatCounter 1999-2021 – Режим доступу до ресурсу: https://gs.statcounter.com/os-market-share.
4. Mobile Phones [Електронний ресурс] – Режим доступу до ресурсу: https://www.statista.com/outlook/13610/100/mobile-phones/worldwide.
5. Object Management Group. Undefied Modeling Language (OMG UML) / Object Management Group., 2017. – 754 с.
6. ISO 9241-210:2010 [Електронний ресурс] // ISO – Режим доступу до ресурсу: https://www.iso.org/standard/52075.html.
7. The State of Mobile in 2020 [Електронний ресурс] // 2021 App Annie – Режим доступу до ресурсу: https://www.appannie.com/en/insights/market-data/state-of-mobile-2020/?sfdcid=7016F000002MS1c&utm_campaign=ww-logo-201910-1910-digital-2020-partnership&utm_content=report-&utm_medium=partnership&utm_source=digital-2020.
8. Unity download [Електронний ресурс] // Unity Technologies – Режим доступу до ресурсу: https://unity3d.com/.
9. With you from concept to commercialization [Електронний ресурс] // Unity Technologies. – 2020. – Режим доступу до ресурсу: https://unity.com/solutions/game.

10. Вигерс К. И. Разработка требований к программному обеспечению / Карл И. Вигерс., 2004. – 576 с.
11. ВИКОРИСТАННЯ ТЕСТОВОГО КОНТРОЛЮ ТА ТЕСТОВИХ ЗАВДАНЬ У НАВЧАННІ ІНОЗЕМНОЇ МОВИ СТУДЕНТІВ НЕМОВНИХ ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДІВ / Т. В. Саварин, І. Р. Бекус. – 2012.
12. Разработка мобильных приложений от А до Я: полный гайд [Електронний ресурс] // © 2021 DAN IT Education. – 2020. – Режим доступу до ресурсу: https://dan-it.com.ua/razrabotka-mobilnyh-prilozhenij-ot-a-do-ja-polnyj-gajd/.
13. ТЕСТУВАННЯ ЯК ЕФЕКТИВНИЙ ІНСТРУМЕНТ ВИМІРЮВАННЯ РІВНЯ ЗНАНЬ СТУДЕНТІВ [Електронний ресурс] / Б. С. Гриник, О. Г. Пилипів. – 2013. – Режим доступу до ресурсу: http://webcache.googleusercontent.com/search?q=cache:ywbcpiBsDuAJ:irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe%3FC21COM%3D2%26I21DBN%3DUJRN%26P21DBN%3DUJRN%26IMAGE_FILE_DOWNLOAD%3D1%26Image_file_name%3DPDF/Nzspp_2013_3_22.pdf+&cd=1&hl=ru&ct=clnk&gl=ua

ДОДАТОК А

```
Скрипт Unity | Ссылок: 0
public class CubeOnPlatform : MonoBehaviour
{
    Rigidbody2D rb;
    Сообщение Unity | Ссылок: 0
    private void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }
    Сообщение Unity | Ссылок: 0
    private void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "MovePlatform")
        {
            this.transform.parent = collision.transform;
        }



    }
    Сообщение Unity | Ссылок: 0
    private void OnCollisionExit2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "MovePlatform")
        {
            this.transform.parent = null;
        }


    }
    Сообщение Unity | Ссылок: 0
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "Player")
            rb.mass = 50;
    }
    Сообщение Unity | Ссылок: 0
    private void OnTriggerExit2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "Player")
            rb.mass = 1;
    }
}
```

```
9   public class QuestionRandom : MonoBehaviour
10  {
11      public GameObject winSound;
12      private GameObject cloneWinSnd;
13      public QuestionList[]questions;
14      public Text[] answersText;
15      public Text qText;
16      List<object> qList;
17      QuestionList crntQ;
18      int randQ;
19      public GameObject moveController;
20      private Stats target;
        Сообщение Unity | Ссылок: 0
21      public void Start()
22      {
23          moveController.SetActive(false);
24          qList = new List<object>(questions);
25          questionGenerate();
26          target = moveController.GetComponent<Stats>();
27          Time.timeScale = 0f;
28      }
        ссылка: 1
29      void questionGenerate()
30      {
31          randQ = UnityEngine.Random.Range(0, qList.Count);
32          crntQ = qList[randQ] as QuestionList;
33          qText.text = crntQ.question;
34          List<string> answers = new List<string>(crntQ.answers);
35          for (int i = 0; i < crntQ.answers.Length; i++)
36          {
37              int rand = UnityEngine.Random.Range(0, answers.Count);
38              answersText[i].text = answers[rand];
39              answers.RemoveAt(rand);
40          }
41      }
        Ссылок: 0
42      public void AnswerBtns(int index)
43      {
44          if (answersText[index].text.ToString() == crntQ.answers[0])
45          {
46              Time.timeScale = 1f;
47              this.gameObject.SetActive(false);
48              Invoke("MoveController", 1f);
49              target.TrueAnswer();
50              cloneWinSnd = Instantiate(winSound) as GameObject;
51              Destroy(cloneWinSnd, 2f);
52          }
53          else
54          {
55              target.FalseAnswer();
56          }
57      }
```

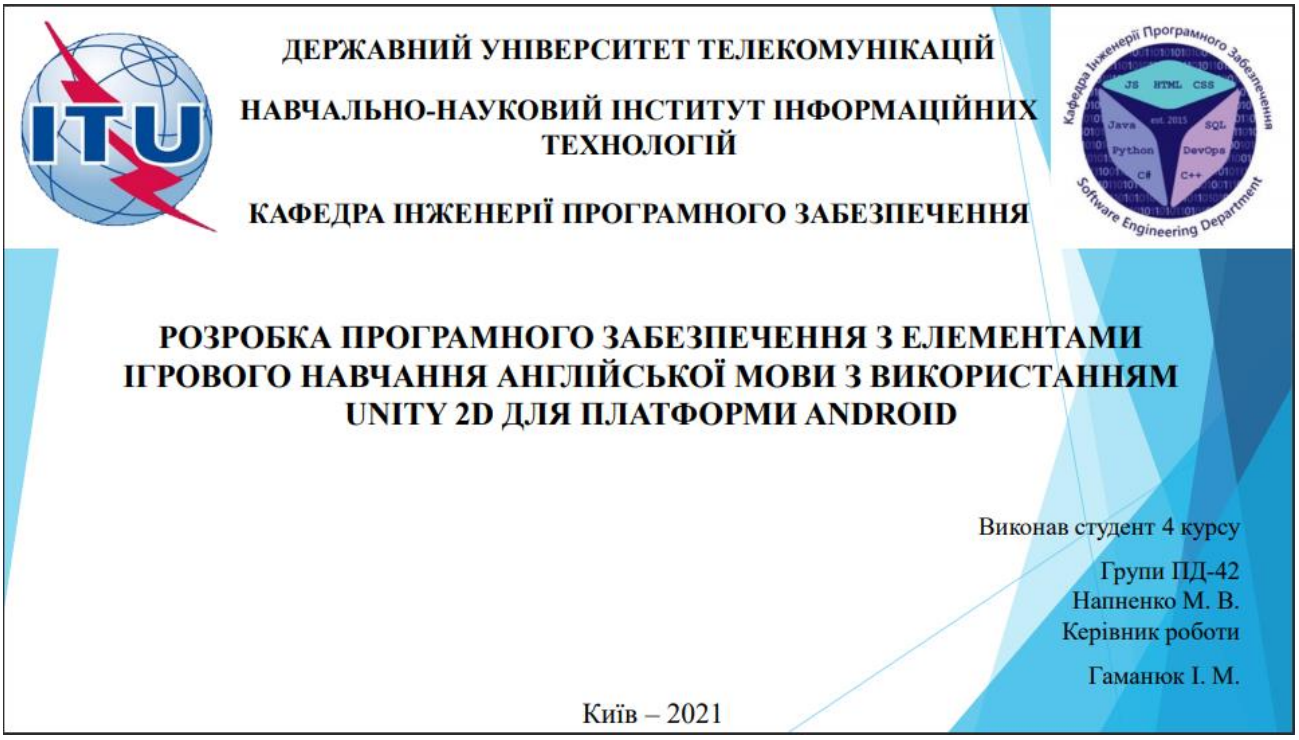
```
public class MovingSpike : MonoBehaviour
{
    public Transform startMovePos;
    public float speed;
    private Rigidbody2D rb;
    public GameObject spikeDestruction;
    private GameObject cloneDestruction;
    public GameObject deadPlayerEffect;
    private GameObject cloneEffect;
    public DeadPlayer deadplayer;
    Сообщение Unity | Ссылок: 0
    private void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }
    Сообщение Unity | Ссылок: 0
    private void Update()
    {
       RaycastHit2D hitinfo = Physics2D.Raycast(startMovePos.position, startMovePos.up);
        if (hitinfo.collider.gameObject.tag == "Player")
            rb.gravityScale = 0.5f;

    }

    Сообщение Unity | Ссылок: 0
    public void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "Player")
        {
            cloneEffect = Instantiate(deadPlayerEffect, rb.transform.position, rb.transform.rotation) as GameObject;
            Destroy(collision.gameObject);
            Destroy(gameObject);
            Destroy(cloneEffect, 1.5f);
            deadplayer.Invoke("Restart", 1.5f);
        }
        else if (collision.gameObject.tag == "Wall")
        {
            cloneDestruction = Instantiate(spikeDestruction, rb.transform.position, rb.transform.rotation) as GameObject;
            Destroy(gameObject);
            Destroy(cloneDestruction, 1.5f);
        }
        else if (collision.gameObject.tag == "IceBlock")
        {
            cloneDestruction = Instantiate(spikeDestruction, rb.transform.position, rb.transform.rotation) as GameObject;
            Destroy(gameObject);
            Destroy(collision.gameObject);
            Destroy(cloneDestruction, 1.5f);
        }
    }

}
```

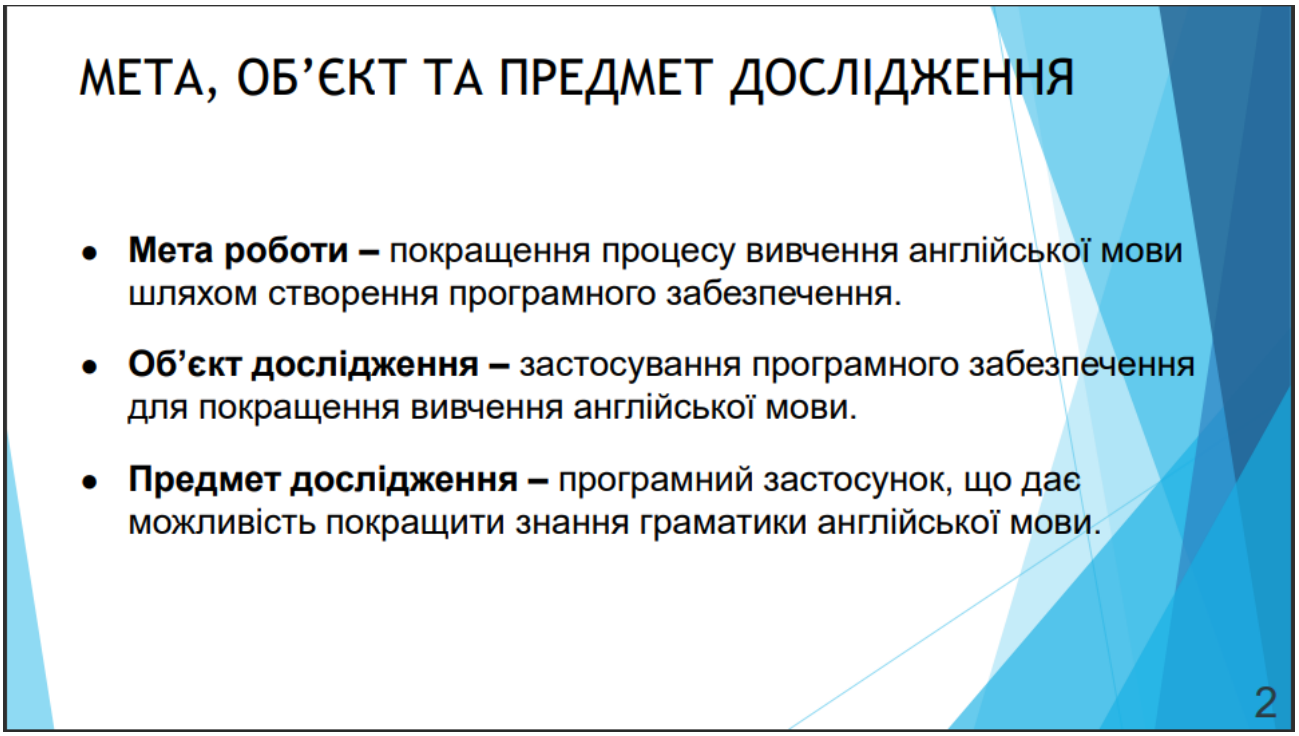
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;


public class PlatformMove : MonoBehaviour
{

    public float speed;
    public float rightRange;
    public float lefttRange;

    bool moveRight = true;



    private void Update()
    {
        if (transform.position.x > rightRange)
        {
            moveRight = false;
        }
        else if (transform.position.x < lefttRange)
        {
            moveRight = true;
        }
        if(moveRight)
        {
            transform.position = new Vector3(transform.position.x + speed* Time.deltaTime, transform.position.y,transform.position.z);
        }
        else
        {
            transform.position = new Vector3(transform.position.x -speed * Time.deltaTime, transform.position.y,transform.position.z);
        }

    }

}
```

```
Скрипт Unity | Ссылок: 0
public class Manager : MonoBehaviour
{
    Сообщение Unity | Ссылок: 0
    private void Start()
    {
        AudioListener.volume = PlayerPrefs.GetInt("sound");
    }
    Сообщение Unity | Ссылок: 0
    void Update()
    {
        if (Application.platform == RuntimePlatform.Android)
        {
            if (Input.GetKeyDown(KeyCode.Escape))
            {
                PlayerPrefs.Save();
                Application.Quit();
            }
        }
    }
    Ссылок: 0
    public void Pause()
    {
        Time.timeScale = 0;
    }
    Ссылок: 0
    public void Resume()
    {
        Time.timeScale = 1;
    }
    Ссылок: 0
    public void Restart()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
        Time.timeScale = 1;
    }
    Ссылок: 0
    public void Quit()
    {
        PlayerPrefs.Save();
        Application.Quit();
    }
    Ссылок: 0
    public void Play()
    {
        SceneManager.LoadScene(1);


    }
    Ссылок: 0
    public void MainMenu()
    {
        SceneManager.LoadScene(0);
        Time.timeScale = 1;
    }
}
```

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

public class SaveLevels : MonoBehaviour
{

    int levelUnlock;
    public Button[] buttons;
    private void Start()
    {

        levelUnlock = PlayerPrefs.GetInt("levels",1);
        for (int i = 0; i < buttons.Length; i++)
            buttons[i].interactable = false;
        for (int i = 0; i < levelUnlock; i++)
        {
            buttons[i].interactable = true;

        }

    }
    public void loadLevel(int levelIndex)
    {
        SceneManager.LoadScene(levelIndex);
    }
    public void NewGame()
    {
        PlayerPrefs.DeleteAll();
        PlayerPrefs.SetInt("levels", 1);
        PlayerPrefs.Save();
    }
}
```

ДОДАТОК Б


ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ЕЛЕМЕНТАМИ ІГРОВОГО НАВЧАННЯ АНГЛІЙСЬКОЇ МОВИ З ВИКОРИСТАННЯМ UNITY 2D ДЛЯ ПЛАТФОРМИ ANDROID
Виконав студент 4 курсу
Групи ПД-42
Напненко М. В.
Керівник роботи
Гаманюк І. М.
Київ – 2021


МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ
• Мета роботи – покращення процесу вивчення англійської мови шляхом створення програмного забезпечення.
• Об'єкт дослідження – застосування програмного забезпечення для покращення вивчення англійської мови.
• Предмет дослідження – програмний застосунок, що дає можливість покращити знання граматики англійської мови.
2

АКТУАЛЬНІСТЬ

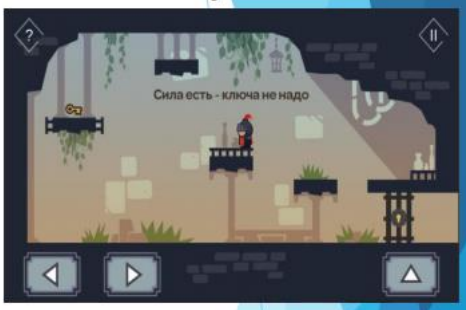
- Популярність мобільних платформ;
- Потреба в вивченні англійської мови учнями та студентами;
- Покращення процесу вивчення англійської мови;
- Брак часу для навчання через використання мобільних девайсів.

АНАЛОГИ

Kub-puzzle platformer

Twinbotz

Tricky castle



Порівняння аналогів та ПЗ

Назва	Локалізація	Масштабованість інтерфейсу	Наявність онлайн магазину	Орієнтація екрану	Елемент головоломки	Елемент навчання
PlayLearn	Англійська, Українська	Так	Ні	Landscape	Так	Так
Kub-Puzzle Platformer	Англійська	Так	Так	Landscape	Так	Ні
Twinbotz	Англійська	Так	Ні	Portrait	Так	Ні
Tricky Castle	Англійська, Російська	Так	Так	Landscape	Так	Ні

ТЕХНІЧНІ ЗАВДАННЯ

- *1. **Назва:** «PlayLearn»;*
- *2. **Призначення:** Програмне забезпечення для покращення процесу вивчення англійської мови;*
- *3. **Область використання:** Школярі та студенти ВНЗ.*
- *4. **Платформа:** «Android»;*
- *5. **Розповсюдження:** магазин застосунків «Google Play Market» на безкоштовній основі*

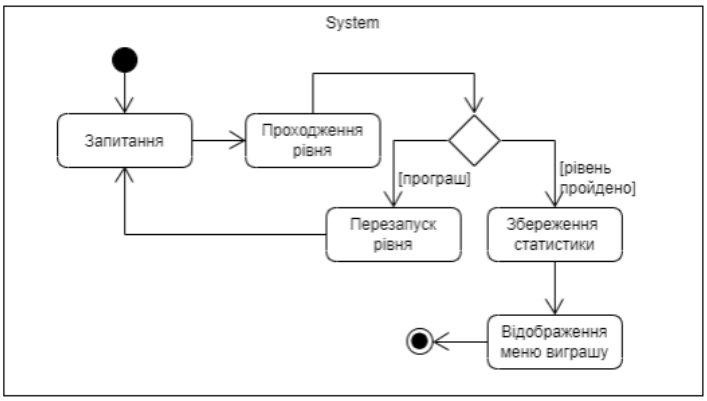
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

1. Ігровий рушій «Unity»;
2. Мова програмування «C#»;
3. Середовище розробки «Microsoft Visual Studio».

ГОЛОВНЕ МЕНЮ

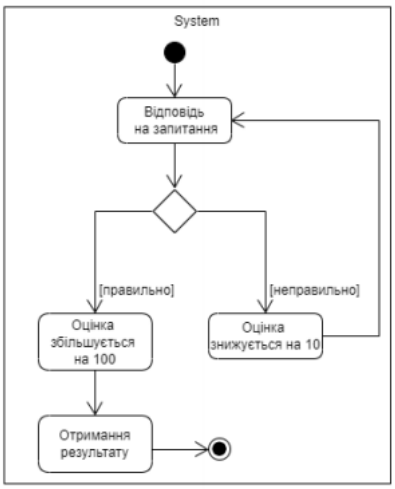
UML-діаграма варіантів використання

ПРОХОДЖЕННЯ РІВНЯ



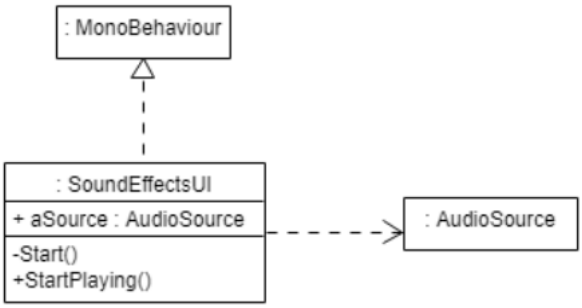
UML-діаграма діяльності

ВІДПОВІДЬ НА ЗАПИТАННЯ ТЕСТУ



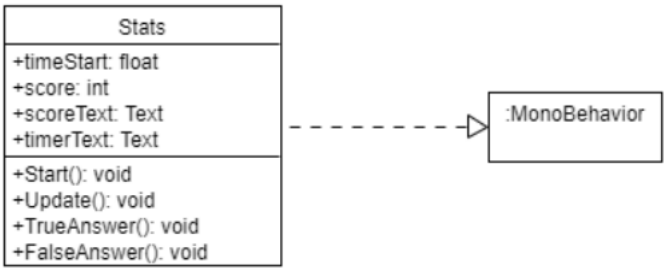
UML-діаграма діяльності

ДІАГРАМА КЛАСІВ



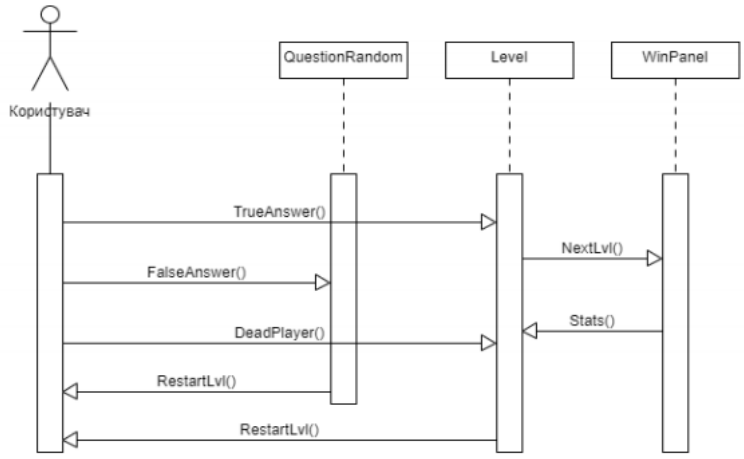
Клас «SoundEffectsUI»

ДІАГРАМА КЛАСІВ



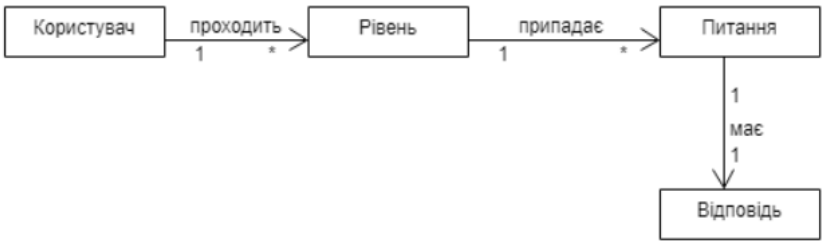
Клас «Stats»

ПОСЛІДОВНІСТЬ СЦЕНАРІЮ ПРЕЦЕДЕНТА «КОРИСТУВАЧ»



Діаграма послідовності

ДІАГРАМА ПРЕДМЕТНОЇ ГАЛУЗІ



Зв'язки:
один (1) Користувач проходить багато(*) Рівнів
на один (1) Рівень припадає багато(*) Питань
одне (1) Питання має одну(1) відповідь.

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

- Напненко М. В. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ЕЛЕМЕНТАМИ ІГРОВОГО НАВЧАННЯ АНГЛІЙСЬКОЇ МОВИ З ВИКОРИСТАННЯМ UNITY 2D ДЛЯ ПЛАТФОРМИ ANDROID / М. В. Напненко. – Київ, 2020: Матеріали всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інфокомунікаційних технологіях». Збірник тез. 05.02.2020, ДУТ, м. Київ — К.: ДУТ, 2020. — С. 83.

ВИСНОВКИ

Результатом роботи є розроблений застосунок, який покращить процес вивчення англійської мови. Проаналізовано ринок мобільних пристроїв та програмного забезпечення. Розглянуті інструменти для розробки ПЗ, їх переваги та недоліки.
- відпрацьовано модель прецедентів (прецеденти (історії користувача), діаграми варіантів використання, діаграма послідовності);
- відпрацьовано бізнес модель (діаграма предметної галузі);
- відпрацьовано модель проектування (діаграми діяльності, діаграми класів);
- реалізовано програмне забезпечення (діаграма артефактів, діаграма розгортання, код (у додатку)).

В подальшому застосунок може бути опублікований в магазині мобільних застосунків «Google Play Market». Буде здійснюватись підтримка та розроблятись оновлення ПЗ.

Пропозиції

Знання, які були отримані в процесі виконання роботи можуть бути використані:

- Для розробки ПЗ під мобільні платформи;
- При проектуванні вимог до системи;
- Для аналізу ринку та можливих програмних рішень.

ДЯКУЮ ЗА УВАГУ!