Національний університет «Полтавська політехніка імені Юрія Кондратюка»

(повне найменування вищого навчального закладу)

Навчально науковий інститут інформаційних технологій та робототехніки

(повна назва факультету)

Кафедра комп’ютерних та інформаційних технологій і систем

(повна назва кафедри)

**Пояснювальна записка**

**до дипломного проекту (роботи)**

бакалавра

(рівень вищої освіти)

на тему:

Розроблення двовимірної комп’ютерної гри у жанрі головоломки

Виконав: студентка4 курсу, групи 401–ТН

спеціальності

122 Комп’ютерні науки

(шифр і назва напряму)

Смірнова О.А\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Керівник Фурсова Н.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Рецензент Пусан А.Ф.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Полтава – 2021 року

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**«ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»**

**НАВЧАЛЬНО НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ**

**КАФЕДРА КОМП’ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ**

**КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА**

**спеціальність 122 «Комп’ютерні науки»**

**на тему**

**«Розроблення двовимірної комп'ютерної гри у жанрі головоломки»**

**Студентки групи 401-ТН Смірнової Олександри Артемівни**

Керівник роботи

кандидат економічних наук,

доцент Фурсова Н.А.

Завідувач кафедри

кандидат технічних наук,

доцент Головко Г.В.

Полтава – 2021

**РЕФЕРАТ**

Кваліфікаційна робота бакалавра: 64 с., 33 рисунка, 1 додаток, 26 джерел.

**Об’єкт дослідження**: проектування та розробка двовимірної комп’ютерної гри у жанрі головоломки з використанням ігрового рушію Unity.

**Мета роботи**: розроблення ігрових об’єктів та механік для гри у жанрі «Головоломка» для реалізації її в якості безкоштовної розважально-навчальної програми для аудиторії молодшого віку.

**Методи**: аналізу і синтезу, мозкового штурму, проектування та розробки комп’ютерної гри.

**Ключові слова**: комп’ютерна гра, головоломка, Unity, двовимірна гра, ігрові об’єкти.

**ABSTRACT**

Bachelor's qualification work: 64 pp., 33 drawings, 1 addition, 26 sources.

**Object of research:** design and development of a two-dimensional computer game in genre of puzzles using the Unity game engine.

**Purpose:** to develop game assets and a mechanics for the game in the genre of "Puzzle" to implement it as a free entertainment and educational program.

**Methods:** analysis and synthesis, brainstorming, computer game design and development.

**Keywords:** computer game, puzzle, Unity, two-dimensional game, game assets.

**ЗМІСТ**

ВСТУП 6

РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ 7

1.1 Аналіз предметної області 7

1.2 Призначення та класифікація комп’ютерних ігор за жанрами 10

1.3 Переваги та недоліки існуючих ігор у жанрі «Головоломка» 17

1.4 Постановка завдання 22

РОЗДІЛ 2 ПРОЕКТУВАННЯ КОМП’ЮТЕРНОЇ ГРИ 24

2.1 Вибір ігрового рушію для проектування 24

2.2 Етапи розробки комп’ютерної гри 27

РОЗДІЛ 3 РОЗРОБКА ДВОВИМІРНОЮ КОМ’ПЮТЕРНОЇ ГРИ У ЖАНРІ ГОЛОВОЛОМКИ 29

3.1 Розробка ігрових ресурсів та їх імпорт 29

3.2 Розробка головного меню 34

3.3 Розробка основної гри 38

РОЗДІЛ 4 ТЕСТУВАННЯ 45

4.1 Вибір виду тестування 45

4.2 Тест план 45

ВИСНОВКИ 49

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 51

ДОДАТОК А ВИХІДНІ КОДИ ПРОГРАМНОГО ЗАСОБУ 54

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ**

**ПК** – персональний комп’ютер.

**Геймплей** – ігровий процес з набором правил для взаємодії гравця з грою.

**Гравець** – особа, яка бере участь у грі.

**ТЗ** – технічне завдання.

**DirectX**– утиліта для роботи з візуальними ефектами, пов’язана з програмуванням ігор.

**DnD** (Drag and Drop) – форма з використанням комп’ютерної миші для можливості виконання певних дій і графічному інтерфейсі.

**ПЗ**– програмне забезпечення.

**UI** – набір інструментів для розробки у користувацькому інтерфейсі.

**ВСТУП**

Технології впродовж багатьох років стрімко розвиваються у різних сферах нашого життя, проникаючи в усі куточки земної кулі. Цей розвиток не оминув і сферу розваг, зокрема – ігри на ПК, які, завдяки розповсюдженню, вже багато років є невід’ємною частиною дозвілля широкого загалу.

Комп’ютерні ігри є чимось унікальним, вони – пряме продовження програм, які не мали нічого спільного з розвагами. Закінчений у своїй суті продукт з передбачуваними ходами та правилами. І саме особливість постанови задачі відокремлює цю програму від інших, а також її творчий потенціал.

Двовимірні ігри займають особливе місце в індустрії. Навіть через графічний скачок та популяризацію тривимірних ігор серед великих студій, більш проста графіка не втратила своєї значимості серед людей, та частіше займає своє місце серед незалежних інді-розробників.

Серед ігор жанр «Головоломка» популярний, проте, лише деякі його різновиди. Такий піджанр, як ігри на пам'ять, не знайшли високої популярності у собі через складність. Проте, це не означає, що він не вартий уваги. Він ближче торкається теми навчальних ігор та добре почуває себе в цьому середовищі.

Мета дипломної роботи полягає в створені гри, яка буде здатна заохочувати гравця концентруватися на проходженні головоломок через систему нагороди, що поліпшить його увагу через наявність мотивації та зафіксує позитивний досвід навчання.

**РОЗДІЛ 1** **АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ**

**1.1 Аналіз предметної області**

Комп’ютерна гра – представлена для гри на персональному комп’ютері програма, призначенням якої є розваги та навчальний процес у розважальній формі. Термінологія відноситься значною мірою до ігор, встановлених на персональному комп’ютері, проте більш широкий термін – відеогра, який застосовується у всіх випадках, коли для відображення контенту гри необхідний екран [1].

Це відокремлений випадок програми, який розцінюється як твір, проте законом охороняється лише його вихідний та об’єктний код [2]. Код є своєрідною інструкцією для комп’ютера, яку він здатний зчитати та зрозуміти. Так він відображає правильний набір та послідовність дій, який корелюється, за його наявністю, вибором гравця.

Незважаючи на своє технічне походження та складну реалізацію, для більшості людей комп’ютерна гра не більше, ніж аудіо-візуальна розвага, особливо сьогодні, коли існує тісна інтеграція кіномистецтва у галузі ігор і такі речі, як сценарій та сюжет, мають значну роль у комплексному створені гри.

Головною характеристикою виділяється простота сприйняття та цікаве, яскраве зображення; існуючі елементи випробувань та змагань приваблюють гравця, простота сприйняття не перевантажує мозкову діяльність, посилює інтерес і бажання продовжувати взаємодію з грою. Відволікаючись на ігровий сюжет, людина здатна знижувати свій рівень стресу від повсякденних справ і це посилює відчуття задоволеності від життя [3].

Від початку свого існування до сьогодні у сучасних комп’ютерних ігор поступово з’являлися нові напрямки та жанри, привертаючи до себе увагу нової аудиторії та розширюючи її межі.

Історія комп’ютерних ігор має тісний зв'язок з технологічним прогресом, так як комп’ютерна гра потребує технології, здатні обробляти великі обсяги даних і представляти ці дані візуально. Тож від розвитку технологій залежить і розвиток комп’ютерних ігор; є ймовірність, що дана залежність також значною мірою може впливати на розвиток техніки. Або вони тісно взаємозв’язані один з одним.

Визначення першої комп’ютерної гри залежить від того, що можна вважати нею. Важко визначити переломний момент створення гри, тож є причини розглянути самі ранні варіанти.

Першою інтерактивною електронною грою можна вважати «Розважальний апарат на базі електронно-променевої трубки» [4].

Пристрій було створено Томасом Т. Голдсмітом молодшим і Естлом Рей Манном у 1947 році, він є аналоговим та імітує управління польотом ракети, тобто симуляція (Рис.1.1.1).

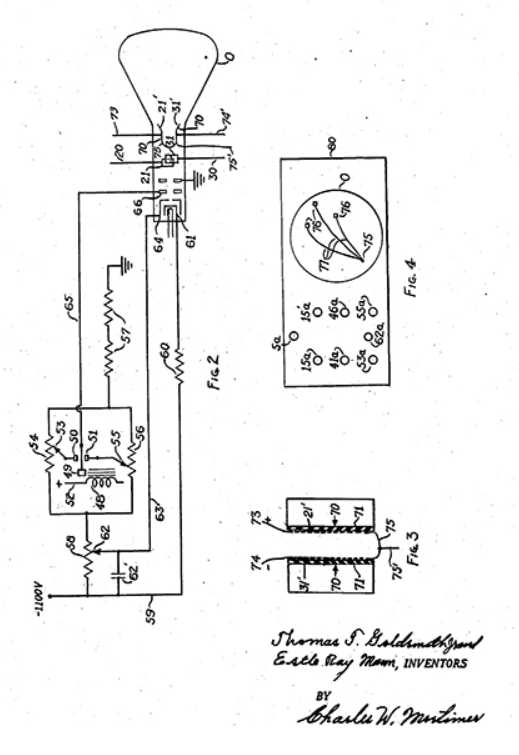


Рисунок 1.1.1 – Схема розважального апарату на базі електронно-променевої трубки

А.С. Дуглас у 1952 році створив гру OXO (Хрестики-нулики), це була одна з самих ранніх комп’ютерних ігор на електронні обчислювальній машині та виведена на дисплей (Рис.1.1.2). Тут відбувалася гра людини та машини.

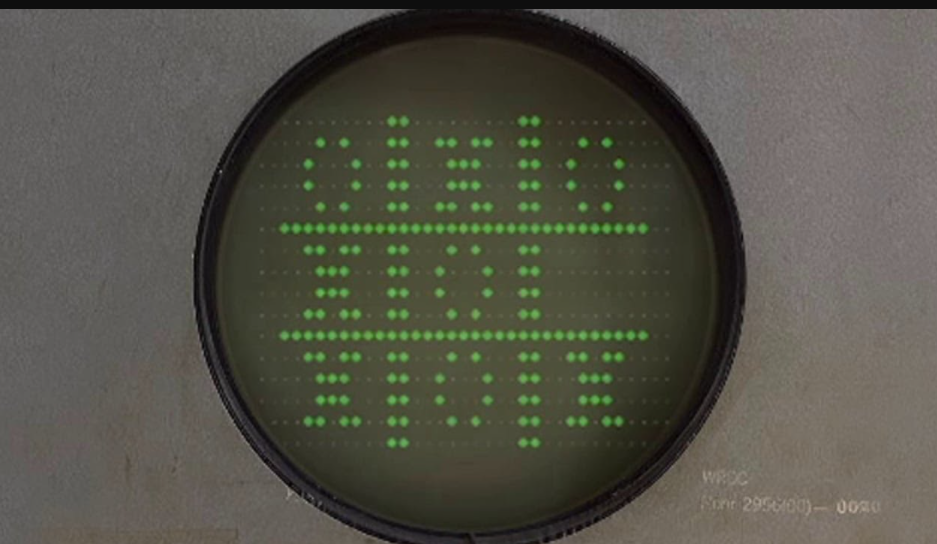


Рисунок 1.1.2 – Гра «OXO»

Більш відомим прикладом однієї з ранніх версій відеоігор є «Теніс-для-двох», яку створив Вільям Гіґінботем у 1958 році [5]. Спортивна відеогра, виділяється тим, що перша мала графічний дисплей (Рис.1.1.3).

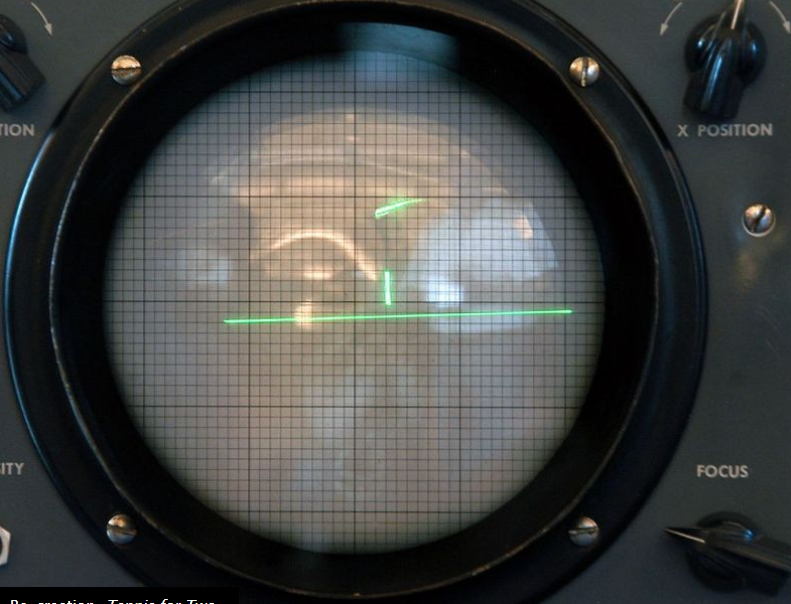


Рисунок 1.1.3 – Спортивна гра «Теніс-для-двох»

Такі ігри створювалися на великих обчислювальних апаратах, вони були важко реалізованими через технічну недосконалість у той час. При тому ігри були максимально прості та невимушеними. Це не відміняє їх значимості для сьогодення, адже без цих важких перших кроків не могло і близько бути того різноманіття, яке є сьогодні.

**1.2 Призначення та класифікація комп’ютерних ігор за жанрами**

Комп’ютерні ігри стали масовим феноменом та їх популярність швидко поширилася за останні десятиліття (так само, як і технічний розвиток). Легше за все інтеграція в світ ігор далася дітям і підліткам, в першу чергу через привабливість простими рухливими картинками та легкістю самої гри. Також велика роль віддається пластичності розуму, якому легше приймати нові речі.

Та більша частина дорослої аудиторії, перша, що зіштовхнулася з масовим явищем комп’ютерних ігор, не могла сприйняти їх як щось, варте уваги. Тож по всьому світу, а особливо серед пострадянських держав, ігри довгий час вважалися чимось шкідливим та навіть небезпечним, особливо для дітей.

Так значення Ґік довгий час мало негативний характер, характеризуючи таку людину як блазня або дивну [6]. Вони часто асоціювалися з технологіями та іграми, проте суспільство довгих час не підтримувало таких людей.

З популяризацією ігор, а також інших атрибутів цієї культури, з простотою доступу до різних жанрів та видів, кількість людей-ґіків почала нестримно рости до ступеня, коли люди, не будучи частиною цієї культури, намагаються наслідувати її, щоб стати частиною популярної спілки.

І сьогодні існує багато стереотипів про шкоду, які наносять відеоігри (комп’ютерні ігри), особливо на дітей.

Не можна сказати, що ігри повністю безпечні, проте залежність від ігор не є причиною проблем, а лише їх наслідок. Тож звинувачення ігор у проблемах доволі часто лише уникнення власної відповідальності за стан дитини.

Усі ігри, не важливо, комп’ютерні чи звичайні, у собі мають дві цілі. Нерівноцінні по значенню, проте обидві важливі, а саме розвага та навчання через імітування реальності.

В першу чергу людина бажає зняти навантаження, в чому допомагають саме комп’ютерні ігри. Не потрібно напружувати мозок надто сильно, процес відносно простий, але з долею виклику, щоб не стало нудно. Можуть бути елементи заохочування, наприклад ігровий предмет або частина історії.

Під час ігрової діяльності відбувається концентрація уваги на змісті сюжету, діючих персонажах. Результати гри дають можливість осмислювати дії, відстежувати прогрес засвоєння навчального матеріалу.

Через гру можливо покращити свої навички, як, наприклад, швидкість реакції або логічне мислення, якщо в грі є механіки, здатні дати подібне навантаження.

З постійним розвитком комп’ютерних ігор з’явився термін гейміфікація, який означає використання ігрових підходів для неігрових процесів [7]. Тож ігри здатні напряму впливати на освітній процес, і не лише у дітей. Мозок не сприймає гру як реальне навчання, тож нова інформація поступає у мозок набагато швидше.

У приклад можна навести факт, що раніше ігри часто використовувалися лише як доповнення до фільмів, книг і музики, коли вони були створені для чогось що вже було популярне, проте поступово саме фільми, книги та музика стали надихатися іграми та використовувати їх популярність для уваги нової аудиторії [8].

Тож ігри можуть використовуватися у багатьох напрямках, вони тісно проникають у різні сфери життя, та їх використовують для полегшення навчального процесу.

У відеоіграх іграх міститься багато різноманітних жанрів і піджанрів [9]. Важко оглянути усі жанри, багато з них працюють окремо один від одного і не поміщаються під одне велике пояснення. Тож я огляну базові та популярні жанри.

**Бойовик**

Бойовик – це жанр ігор, в якому ціниться швидкість реакцій гравця [10]. Такі ігри динамічні, швидкі, зав’язані на високу концентрацію та точність кожного руху.

Він має в собі такі характеристики як: наявність рівнів, які необхідно проходити, деяку кількість життів (або лише одне життя на всю гру), долати перешкоди та ворогів, у кінці зустрінеться з Босом – останнім та найважчим елементом гри.

Проте такий опис підходить до половини усіх існуючих ігор; різниця в тому, що в більшості випадків бойовик використовується з певними доповненнями (як пригодницький бойовик). Назвати гру «чистим» бойовиком можливо в рідких випадках.

Під собою бойовик має такі піджанри:

* платформер – гра, зосереджена на стрибках по платформах;
* шутер – гра, в більшості зосереджена на знищені ворогів за допомогою зброї;
* файтинг – гра, зосереджена на боях один на один в обмеженому просторі;
* побий їх усіх – гра, зосереджена на боях гравця з великою кількістю ворогів на великих площах;
* стелс – це гра, зосереджена на тому, щоб не бути поміченим, ховаючись від ворогів;
* симулятор виживання– гра, зосереджена на збережені життя головного героя у небезпечній ситуації;
* ритмічна гра – гра, зосереджена на музиці та ритму.

**Пригодницькі ігри**

Пригодницькі ігри визначаються за стилем ігрового процесу (геймплею). Тут гравець взаємодіє з своїм оточенням, другими персонажами, гравцями [11]; вирішує головоломки щоб просунутися по сюжету гри. Пригодницькі ігри часто мають всередині додатково міні-ігри. В цілому жанр потребує від гравця роздумів над загадками та уважності в деталях гри.

Під собою пригодницькі ігри мають такі піджанри:

* текстова пригода – гра, зосереджена на текстовому інтерфейсі без використання графіки;
* графічна пригода – гра, зосереджена на візуальних картинках та тексті з можливістю використовувати комп’ютерну мишу як контролер;
* візуальні новели – гра, зосереджена на історії, яка передається через статичні картинки персонажів, тексту діалогів та звуку з рідкою можливістю зробити вибір;
* інтерактивне кіно – гра, зосереджена на показу фільму або мультфільму з можливістю зробити вибір у деяких точках сюжету;
* 3D у реальному часу – гра, зосереджена на 3D грі у реальний час.

**Пригодницький бойовик**

Пригодницький бойовик – це суміш жанрів пригодницьких та бойових ігор, який в першу чергу виділяється поставленим головним завданням на всю гру, всередині цього завдання містяться або завдання поменше, які приводять гравця до виконання основної мети, або колекційні предмети, які виконують схожу функцію [12].

Під собою пригодницький бойовик має такі піджанри:

Пригодницький бойовик – жанр для великої гри, з значною кількістю персонажів, історією світу, кропіткого сюжету та діалогів. Також в його запасі наявні різні види зброї, інвентар, в який необхідно складати знайдені предмети, битви з ворогами різного рівня складності; головоломки та дослідження різних рівнів гри.

Під собою пригодницький бойовик має такі піджанри:

* виживальне жахіття – гра, зосереджена на атмосфері жахів та жорстокості, де головною метою є виживання у цих умовах;
* метроідванія – гра, зосереджена на нелінійній системі, дослідженню гри, відкриванню закритих територій та сюжеті.

**Рольова відеогра**

Рольова гра – один з найпопулярніших жанрів ігор, в якому основою є рольова система, тобто задана сукупність правил та комп’ютерна програма у ролі ведучого гри. Усі рольові ігри в першу чергу були побудовані на основі настільної гри «Dungeons & Dragons» («Підземелля і дракони») [13]. Найчастіше дії таких ігор відбуваються у обстановці середньовіччя або фентезі-світ.

Серед характеристик жанру є такі елементи, як управління персонажем, його розвиток та параметри, наявність зброї та ворогів, прийняття рішень, можливий успіх або програш через прийняття рішень згідно заданих правил.

Під собою рольова гра має такі піджанри:

* рольовий бойовик – гра, зосереджена на змішуванні жанрів «Рольові ігри» та «Бойовик», проходженню гри у реальному часі, залежності проходження гри не тільки від персонажа, але і швидкості гравця;
* багатокористувацька онлайн рольова гра – гра, зосереджена на рольовій онлайн грі з великою кількістю гравців які можуть взаємодіяти один з одним.
* rogue-подібні– гра, зосереджена на випадково-створених рівнях та покращенню персонажа через проходження ворогів та перешкод;
* тактична рольова гра – гра, зосереджена на прийнятті тактичних рішень під час гри з обмеженою кількістю ресурсів проти ворога;
* пісочниця – гра, зосереджена на відкритому світі, взаємодією з оточенням та мінімальних обмеженнях;
* групова рольова гра від першої особи (First-person party-based RPG) – гра, зосереджена на груповій грі від першої особи.

**Симулятор**

Основна ідея ігор-симуляторів – відтворення реальності для її імітації. Від гонок на машинах, спортивних ігор до побудови міст, створення людей, які будуть жити всередині грі та імітувати життя [14].

Під собою ігри симулятори мають такі піджанри:

* симулятор будівництва і автоматизації – гра, зосереджена на будівництві, управлінні ресурсами та проектами;
* симулятор життя – гра, зосереджена на створенні живого об’єкта та симуляції життя з ним;
* симулятор транспортного засобу – гра від першої особи, зосереджена на відтворенні досвіду польоту на літаку або водіння транспортного засобу;
* спортивний симулятор – гра, зосереджена на відтворені досвіду у спортивних іграх різного напряму.

**Стратегічна відеогра**

Стратегічні відеоігри націлені на перемогу за допомогою стратегій та тактик. Дані ігри беруть основу з класичної гри «шахи», які були в першу чергу військовою грою [15]. Тож у стратегічних іграх є суперник і надається обмежена кількість ходів, а потім передається хід іншій стороні. І так по черзі до перемоги одного із гравців.

Другий розповсюджений варіант гри більш реалістичний і дозволяє робити безліч ходів одночасно усім.

Під собою ігри стратегії мають такі піджанри:

* глобальна стратегія – гра, зосереджена на задачах: досліджувати, розширювати, експлуатувати та знищувати;
* артилерія – гра, зосереджена на танкових битвах з елементом шутеру;
* стратегія в реальному часі – гра, зосереджена на виконанні ходів/дій з можливістю зробити це одночасно з суперником, а також управлінні ресурсами;
* тактика в реальному часі – гра, зосереджена на симуляції тактики у військовій тематиці в режимі реального часу;
* багатокористувацька онлайн бойова арена – гра, зосереджена на стратегії у реальному часі з можливістю керувати лише одним персонажем з метою перемогти ворога;
* захист вежами – гра, зосереджена на захисті від ворогів контрольованих комп’ютером;
* покрокова стратегія – гра, зосереджена на здійсненні дій по черзі зі своїм противником без можливості робити це одночасно;
* покрокова тактика – гра, зосереджена на реалістичному відображенні військових тактик та прийманні рішень з обмеженою кількістю ресурсів;
* військова гра – гра, зосереджена на відтворенні стратегічної або тактичної війни на основі карт.

**Головоломка**

Ігри головоломки вміщають в себе мету розважити гравця за допомогою ігор на логіку та принести йому задоволення від вирішення задачі, яка потребує час щоб знайти правильне рішення [16]. Для головоломок необхідно задіяти логічне та стратегічне мислення. Загалом головоломки рідко існують у чистому виді, це представляється простою грою, сконцентрованою на рівнях.

Добре працює у інших іграх у виді піджанру.

Головоломки поділяються на два великих піджанри:

* не фізичні – не потребують взаємодії з оточенням всередині гри;
* фізичні – потребують взаємодії з оточенням та об’єктами всередині гри.

**Інші**

* казуальні відеогри – гра, зосереджена на простоті механік та інтерфейсу для масового споживача;
* клікер – гра, зосереджена на простих завданнях, які виконується через клацання мишею;
* групова гра – гра, зосереджена на групових змаганнях між гравцями;
* навчальна гра – гра, зосереджена на навчальних процесах за допомогою використання ігрових елементі.

**1.3 Переваги та недоліки існуючих ігор у жанрі «Головоломка»**

Для знаходження найкращого рішення для проектування гри, необхідно розглянути існуючі проекти у жанрі «Головоломка».

Для аналізу обрано наступні ігри:

* Fract OSC;
* The Witness;
* Crayon Physics Deluxe.

**Fract OSC**

Являється тривимірною музичною грою головоломкою, створеною незалежними інді-розробниками із студії Phosfiend Systems [17] та випущена 22 квітня 2014 року (Рис.1.3.1).

Гра відкрита та дозволяє гравцю досліджувати абстрактні пейзажі, які містять головоломки, побудовані на використанні платформ та музичних елементів.

Вирішення головоломки створює музику, яка буде грати в навколишньому середовищі продовжений час, а також після кожної такої головоломки буде розблокований музичний елемент, який дозволить створювати власну музику.



Рисунок 1.3.1 – Гра «Fract OSC»

**Переваги**

В першу чергу гра помітна своїм візуальним дизайном: приємні кольорі, які оточують, не напружують око надто сильно, усе доволі мінімалістично. Електронна музика ритмічна та енергійна.

Сам підхід головоломок, пов’язаних на музиці та звуках, став характерною рисою гри, який здатен виділити її поміж інших. Це простий метод відрізняє даний проект поміж інших у своєму жанрі, та не надто популярний через складність у реалізації та потребі ретельно продумувати принципи роботи.

Гра здатна мотивувати гравця за схемою моментальної нагороди та глобальної, тобто при вирішені головоломки гравець одразу отримує щось (отримання музичного елементу, відкриття нового проходу або платформи), а при вирішенні певної кількості головоломок, він може створювати власну музику за допомогою музичних елементів та закінчити гру.

**Недоліки**

Складність деяких рівнів, обумовлена не логічними загадками, а потребою запам’ятовувати велику кількість послідовностей, може роздратувати гравця та змусити вимкнути гру.

Велика проблема у монотонності під час проходження. Дослідження світу займає не на довгий час, а потім усе стає одноманітним. Разом з тим рівні становляться важчими та менш цікавими.

У початку гри існує проблема звукового ряду, який не підходить до спокійної гри, звуки раптові та дратівливі, музика на фоні викликає тривогу.

Після завершення головоломки, з’являється музичний супровід, який нагромаджується на музичний фон, який нікуди не зникає, так виникає зайве навантаження на вуха.

**The Witness**

Пригодницька інді-гра, яка концентрує свою увагу на головоломках, створена Джонатоном Блоу [18] у команді Thekla Inc. 26 січня 2016 року (Рис.1.3.2).

У гри відкритий світ острова з безліччю локацій і споруд. Варіативність головоломок у своєму зовнішньому вигляді, взаємодії та складності, більшість з них пов’язана з екранами, на яких необхідно провести лінію певним чином, проте задля їх вирішення необхідно взаємодіяти з оточенням для знаходження підказок.



Рисунок 1.3.2 – Гра «The Witness»

**Переваги**

Великий світ, який цікаво досліджувати; головоломки, пов’язані з предметами гри, які є сюжетно-важливими. Візуально картинка гри проста, але приємна на погляд.

Надзвичайно велика кількість головоломок, але далеко не всі необхідно досліджувати. Залишається великий простір для таємниць, які гравцям буде цікаво знаходити. Так таємниці, які здаються не важливими, привертають до себе увагу та виникає бажання їх розплутати.

Головоломки у грі не одноманітні, а їх складність наростає поступово, тож навчання гравця проходить поступово та приємно.

Історія не подається відкрито, вона дає шанс самостійно поміркувати над нею, знаходячи безліч підказок по всій локації.

**Недоліки**

Спостерігається експлуатація однієї ідеї, яка виконана не ідеально і через це має багато проблем під кінець гри.

Хоч картинка у грі приємна, але на постерах з першого погляду не виділяється серед середнього рівня симуляторів, через це легко пройти повз гри.

Більшість головоломок якимось чином пов’язані з екранами, гра стала заручником цієї ідеї, і хоч би як автори не намагалися зробити велику варіативність головоломок, під кінець вони починають становитися не цікавими та одноманітними.

Велика їх кількість, більше шести сотень на всю гру, робить цю проблему більш помітною. Адже кількість перемагає якість, фізично неможливо зробити велику кількість головоломок якісними.

Гра задає багато питань, проте не на всі основні питання дає відповідь, через що виникає недомовленість, у результаті гравець не задоволений. Через надмірне ускладнення гри під кінець та надмірну кількість питань без відповідей проходити гру самотужки немає бажання.

**Crayon Physics Deluxe**

Арт-гра головоломка, створена інді-розробником Петрі Пурчо [19] 7 січня 2009 року (Рис.1.3.3).

Гра виповнена у стилі простого дитячого малюнку. Її суть на всіх рівнях однакова – кулею, яку можливо підштовхувати, дістатися до зірки, яка присутня на рівні. Окрім найпростішої взаємодії з нею, необхідно малювати фігури за допомогою миші, створюючи об’єкти, які отримують просту фізику всередині гри (падають, взаємодіють з іншими предметами), щоб куля змогла дійти до місця призначення та отримати зірку, щоб перейти на наступний рівень.

**Переваги**

Помітно виділяється поміж інших ігор своїм мінімалістичним, дитячим стилем. Це привертає увагу, викликаючи інтерес.

У грі цікава механіка з можливістю малювати що завгодно, отже різні методи по вирішенню проблеми. Такий підхід здатен розвивати творче мислення та логіку.



Рисунок 1.3.3 – Гра «Crayon Physics Deluxe»

Гра достатньо легка та приємна, в ній можливо самотужки контролювати рівень складності – у ній відсутня точна послідовність дій для вирішення проблеми.

**Недоліки**

Через легкість, навіть на високих рівнях, швидко втрачається мотивація закінчити гру. В ній відсутній сюжет, історія та загадки. Простота на рівні історії в поєднанні з можливістю легко проходити кожен рівень – це поганий приклад створення цікавої гри.

Фізичні властивості намальованих конструкцій знаходяться на мінімальному рівні взаємодії з оточенням. Об’єм та величина ніяк не будуть впливати на об’єкти.

В підсумку, аналізуючи ігри, які концентрують свою увагу на вирішенні головоломок, можна висловити думку про фактори, які визначають якісну гру такого жанру:

* знаходження цікавого способу виділити гру поміж інших;
* поступове підвищення складності, але без зміни суті;
* система нагороди для утримання мотивації гравця;
* візуально приємна картинка самої гри.

Серед типових проблем жанру виділяється складність: без цікавої ідеї гра не буде виділятися, отже не матиме комерційного успіху та підтримки. Отже, необхідно розробити оригінальний продукт, заздалегідь продумуючи деталі гри та послідовність проходження головоломок, а також в яких точках гри слід давати нагороди та у яких кількостях.

**1.4 Постановка завдання**

Після вибору теми, аналізу вже існуючих проектів у жанрі «Головоломка», необхідно сформулювати ТЗ по проектуванню комп’ютерної гри. На виході має бути розроблена двовимірна гра у жанрі головоломки, призначенням якої є розважання гравців та покращення їх пам’яті.

Гра не має під собою мети заробітку, тож буде представлена абсолютно безкоштовно на одній із ігрових платформ.

Комп’ютерна гра – це програмний продукт, та їй потрібно реалізовуватися на ПК користувачів. Тож необхідно визначитися з платформою, на якій буде розміщатися продукт. Windows – одна з самих популярних операційних платформ, і більшість людей користується саме нею.

Гра достатньо проста та не містить складних елементів. Вона орієнтована на масову аудиторію людей, тож її налаштування повинно бути максимально простим.

Мінімальні системні вимоги до гри:

* Операційна система: Windows 7/8/8.1/10
* Процесор: сумарна тактова частота ядер не менше ніж 1.8 гігагерців
* Аудіокарта: Intel HD або вище
* Відеокарта: GeForce 9100 або вище (сумісна з DirectX 9.0)
* DirectX: версія 9.0
* Оперативна пам'ять: 512MB
* Жорсткий диск: 700 MB вільного простору

**РОЗДІЛ 2** **ПРОЕКТУВАННЯ КОМП’ЮТЕРНОЇ ГРИ**

**2.1 Вибір ігрового рушію для проектування**

Після оформлення ТЗ необхідно визначитися з інструментарієм гри. Проведемо аналіз ігрових рушіїв, так як саме в них і буде створюватися наш проект.

Ігровий рушій представлений у вигляді програми, використовується для розробки відеоігор із використанням специфічних прийомів у програмуванні та інструментів, що відповідають технічним вимогам у графіці, звуку, музики, фізичного моделювання або штучного інтелекту [20].

Він включає в себе інтерфейс для створення та програмування, а також ряд програмних бібліотек, які забезпечують високоякісну графіку та візуалізацію, моделювання фізики реального світу, механізми взаємодії для впровадження цифрових інтерактивних ігор для різних платформ.

Існує велика кількість ігрових рушіїв, придатних для проектування комп’ютерних ігор. Вони дуже різняться між собою за кількістю функціоналу, можливостей та ціною.

Найпопулярнішими професійними рушіями по сьогодні є Unity та Unreal Engine. Їх переваги у наявній безкоштовній версії та значному функціоналу, проте вони важкі у опануванні. Проте є відмінності, на які потрібно опиратися при виборі платформи

**Unity** – популярний ігровий рушій від компанії UnityTechnologies. На ньому розробляються трьох- та двовимірні ігри, в подальшому здатні працювати на великій кількості платформ (як наприклад Windows, Android, Apple, iOS, Linux, PlayStation та Xbox).

Серед можливостей рушію присутній адаптивний інтерфейс, ігри пишуться всередині сцен (рівнів), відокремлених файлів всередині проекту. У Unity працює підтримка фізики об’єктів для точної імітації реальних об’єктів. Наявний рендеринг об’єктів та скрипти, за допомогою яких можливо задати сценарій поведінки у проекті. Основна мова у Unity, що підтримується, C#.

Рушій має дві версії: PersonalEdition та ProfessionalEdition. Перша надається безкоштовно і нею можливо користуватися до досягнення певного доходу. Друга версія платна і має незначний додатковий функціонал, більш важливий для розроблення великих тривимірних ігор з високою деталізацією.

Лише одна версія вважається професійною, проте багато інді-розробників вільно використовують безкоштовну версію та створюють великі деталізовані проекти [21].

**Unreal Engine** – ще один потужний рушій, розроблений компанієюEpicGames. Він безкоштовний, проте з важним уточненням, що дохід гри не становить певну суму, інакше необхідно платити компанії. На відміну від Unity, різниця безкоштовної та «платної» версії повністю відсутня [22].

Рушій працює з більшістю платформ, як і Unity до цього. Основою мовою являється С++ та має функцію CodeView, яка спрощує редагування коду. Рушій відрізняється сильною базою інструментів для розробки ігор, що дозволяє створювати фотореалістичні ігри, що більш важливо для тривимірних проектів.

Unreal Engine адаптивний до різних модифікацій, що дозволяє здобути високий рівень унікальності ігрового проекту. Програма має більшу цінність для людей, які працюють з тривимірними проектами.

Існує багато менш популярних аналогів. Під собою вони мають не саму сильну базу інструментарію, натомість більш доброзичливі до початківців. Так їх головна перевага та недолік у простоті: платформи націлюються на людей, яким важко працювати з кодом і у них відсутня можливість найняти іншу людину.

Дається швидке освоєння програми та створення гри без зайвих зусиль, натомість існує велика кількість обмежень через оптимізацію роботи з кодом. Легкість компенсується складністю відходити від певних шаблонів під час роботи.

Розглянемо декілька аналогів популярних платформ.

**GameMaker: Studio** в першу чергу виділяється орієнтацією на двовимірні ігри, рушій адаптований під їх проблеми: він дозволяє додавати у гру власні об’єкти та редагувати їх всередині рушію. Розробником виступає YoYo Games [23].

Він також багатоплатформний, тобто так само, як і популярні рушії, може створювати та адаптувати ігри під різні види платформ, що є великим плюсом для розробників.

Всередині рушію вбудований магазин зі спрайтами, анімаціями, музикою та іншим. Це допомагає швидко знайти забраклий елемент для гри та впровадити його в проект.

Рушій не потребує обов’язкового знання мови програмування, дає можливість новачкові вільно війти у середу розробки. Через графічну середу розробки DnD можливо працювати людям, які більш художньо-орієнтовані.

Для професійних користувачів існує власний язик програмування, написаний на Delphi. Існує безкоштовна версія з обмеженим інструментарієм, яка сильно відрізняється від платної версії.

**Godot**– це перший у списку повністю безкоштовний рушій для дво- та тривимірних ігор, розроблений у співавторстві Хуана Лінетськи та Аріель Манзур [24].

Інструментарій всередині рушія виділяється простотою в освоєнні для початківців та мінімальним застосуванням коду. У ньому існує нодова система, яка автоматизує процес створення гри, та в більшості потребує правильно послідовність дій на схемі.

Godot багатоплатформений, як і інші представлені рушії. Це показник важливості представлення гри на різних платформах.

У грі можливо використовувати дві мови програмування: C++ та GDScript (власна мова для скриптування у рушію).

Вибір залишається на рушію Unity, так як у порівнянні з іншими він має безкоштовну повну версію, великий інструментарій та мову програмування C#

**2.2 Етапи розробки комп’ютерної гри**

Перед початком гри створюється її план, щоб в подальшому робота проходила швидко та без білих плям, які можуть завадити при роботі.

Створення гри можна поділити на чотири основних етапи [25]:

* концепція;
* препродукція;
* продукція;
* постпродукція.

**Концепція**

Необхідне визначення головної ідеї гри, та яку мету вона за собою несе. Якщо можна відповісти на ці запитання, тоді можливо відштовхуватися від них при створені гри. Наприклад, визначення жанрів, на яку аудиторію необхідно опиратися при створені, вибір механік.

Ідея гри йде навколо пташеня, яке не пам’ятає, що з ним трапилося. Є чітка мотивація закінчити гру для гравця – дізнатися, що трапилось. за допомогою гри «Саймон». Гра має набір фігур з різними кольорами, які необхідно натискати у певному порядку. Це виступає як симуляція зусилля пташеня згадати події минулого. Проект націлений на молодшу аудиторію.

**Препродукція**

Або ж «Підготовчий період», під час якого ведеться робота над більш детальним планом гри. Можуть вноситися корективи до початкової концепції, якщо в ній є непрацюючі елементи, що погіршують проект.

Розробляється дизайн гри та ігрові ресурси (фони, малюнки, анімації та інше). Паралельно іде робота над вибором ПЗ та технологією створення ігрового проекту. Гра буде розроблятися на рушію Unity у двовимірному стилі.

**Продукція**

Або ж виробництво. Тут все зводиться до різних етапів роботи по створенню гри, від написання скриптів до створення механік, роботи в більшості випадків ведуться паралельно. Також до проекту додається звук і музика, якщо вони необхідні.

Розробляється діаграма діяльності для розуміння логіки роботи гри (Рис. 2.2.1).

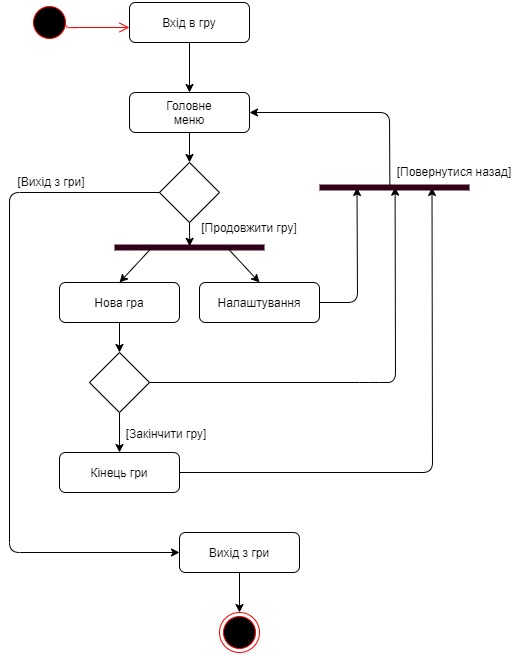


Рисунок 2.2.1 – Діаграма діяльності гри

**Постпродукція**

На постпродукції відбувається етап тестування. На ньому необхідно виявити усі можливі помилки. Для списку знайдених помилок розробляється рішення. Під кінець проводяться додаткові налаштування гри.

**РОЗДІЛ 3** **РОЗРОБКА ДВОВИМІРНОЮ КОМ’ПЮТЕРНОЇ ГРИ У ЖАНРІ ГОЛОВОЛОМКИ**

**3.1 Розробка ігрових ресурсів та їх імпорт**

Для візуалізації гри необхідно створити ігрове оточення. Воно складається з ігрових ресурсів та може бути знайдене у вбудованому магазине Unity, або створене власноруч.

Для створення ігрових ресурсів необхідні дві програми:

* Adobe Photoshop – платформа для графічного редагування;
* DragonBones – платформа для створення анімації.

**Розробка та анімація персонажу.** Усе починається з персонажу. У грі відпадає необхідність, щоб він пересувався з місця на місце, проте для більш красивої, живої картинки необхідно зробити йому просту анімацію гойдання. Стиль гри та його аудиторія визначені – простий та приємний дизайн для молодшої групи людей, якій цікаво бачити приємно намальованого героя, якому хочеться співчувати.

Робота починається зі створення персонажу. Для початку створюється начерк: він окреслює головні форми та допомагає більш точно відтворити його на повноцінному малюнку. Рухому частини тіла малюються на окремих шарах, щоб в подальшому створити більш приємну анімацію (Рис. 3.1.1). Готовий файл імпортується у безкоштовну програму для анімації DragonBones [26] (Рис. 3.1.2).

У DragonBones працює скелетна анімація. Тож для придання рухливості, кожен окремий об’єкт тіла виділяється для надання йому фізичних властивостей. На кожному з них створюються кістки, які прикріпляються до них та роблять тіло рухливим (Рис. 3.1.3).

Для правильної роботи існує основна кістка, яка в подальшому буде рухати усе тіло, і вона розміщена на тулубі. У неї існують дочірні кістки, а у дочірніх кісток можуть бути свої. Це створює логіку анімації.



Рисунок 3.1.1 – Створення пташеня



Рисунок 3.1.2 – Імпорт у Dragon Bones

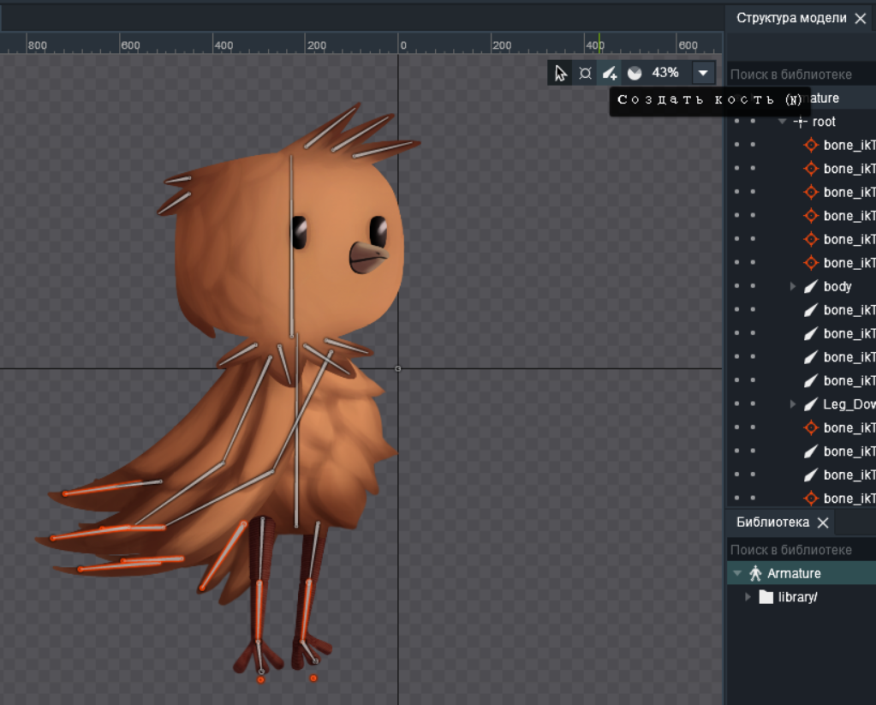


Рисунок 3.1.3 – Створення кісток

Анімація відбувається за технологією ключових кадрів. Тобто необхідно поставити готового персонажа зі скелетом, прикріпленим до нього, у правильну позу, а після робити ключові кадри анімації – найвищі точки зміни тіла (Рис. 3.1.4). Кадри між ними створюються автоматично. Для придання руху реалістичності анімація допрацьовується: між ключових кадрів створюються кадри руху дочірніх кісток.



Рисунок 3.1.4 – Анімація ключовими кадрами

Виведення анімації відбувається через експорт. При цьому нічого змінювати у налаштування експорту не потрібно. Необхідно лише обрати пусту папку для анімації (Рис.3.1.5).

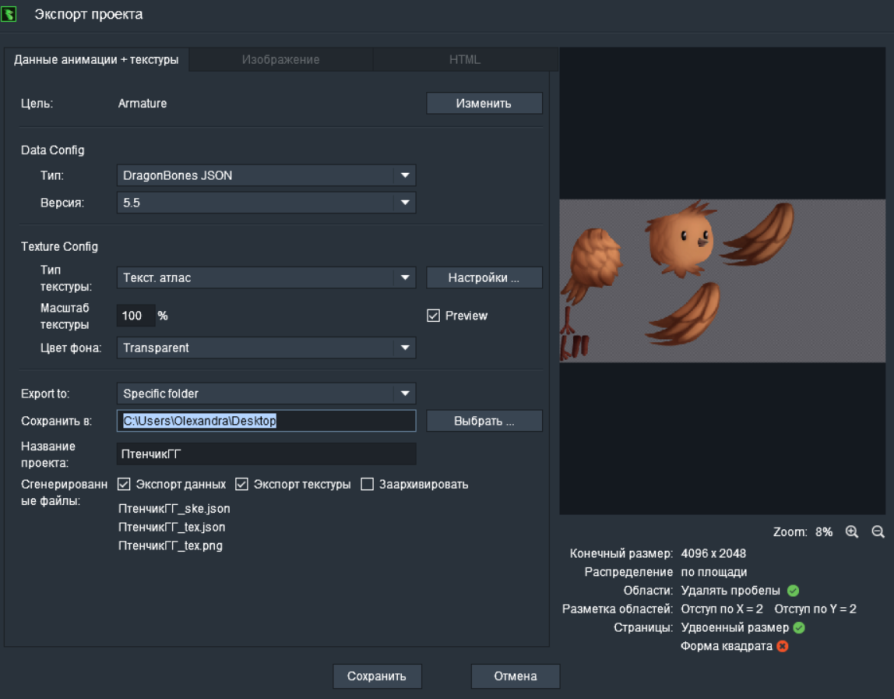


Рисунок 3.1.5 – Експорт проекту

Для імпортування анімації з DragonBones до Unity необхідно встановити у рушій пакет DragonBonesForUnity.unitypackage. Далі він встановлюється через натискання «Assets» у вікні рушію, у ньому обирається «Import New Asset…», а після відкривається нове вікно зі списком файлів, серед яких потрібно знайти встановлений через офіційний сайт пакет.

Файл з анімацією поміщається у одну з папок всередині DragonBones. Йде послідовність дій всередині папки: «Create» – «Dragon Bones» – «Armature Object».

Створюється файл «Armature»,анімаційний об’єкт з’явиться на головному екрані. «Armature» змінюється на «bird».Усі назви всередині сцен будуть англомовні, щоб при створенні скриптів не виникало проблем.

**Звуковий супровід.** Звуковий супровід необхідний переважно для музичного супроводу по всій грі та звуків для гри «Саймон». Для цього були використані звукові доріжки з використанням ліцензії «Royalty Free», що позначає безкоштовне використання супроводу у освітніх та комерційних роботах.

**Статичні ігрові ресурси.** Ігрові ресурси створюють візуальну частину гри, а також для економії часу більша їх частина може бути статичною.

Компенсація статичності відбувається за рахунок деталей всередині грі, які наводять гравця на роздуми про те, що відбувається всередині гри.

У програмі Adobe Photoshop створюємо тло для основної частини гри (Рис. 3.1.6). Тло повністю статичне, та виконує роль естетично-приємної картинки для гравця.

Картина з дівчинкою малюється окремо, так як цей ігровий об’єкт подалі буде заскриптованим (Рис. 3.1.7).

Тло головного меню малюється максимально просто, його задача зведеться до розміщення на собі тексту, створеного всередині рушія. Тож додаткового тексту на даному етапу воно не потребує (Рис. 3.1.8).



Рисунок 3.1.6 – Тло для гри



Рисунок 3.1.7 – Картина «Дівчина»



Рисунок 3.1.8 – Тло головного меню

Останніми ігровими об’єктами слугують нагороди. Вони представлені у виді картинок (Рис. 3.1.9).



Рисунок 3.1.9 – Ігрові об’єкти нагороди

Картинки відповідають за візуальну розповідь історії. Під час основної гри можливо знайти чотири картинки. Остання буде додатковою нагороду і показує фінал гри. Після цієї картинки гра закінчиться.

**3.2 Розробка головного меню**

Створюємо сцену для головного меню, для цього у вікні, де розміщуються сцени, тиснемо правою кнопкою миші, натискаємо на «Create»–«Scene» - називаємо «Menu». Головне меню – це початок гри. З нього можливо почати гри, зробити налаштування, або вийти з гри.

Тож у робочому вікні рушію Unity розміщуємо тло, ігровий об’єкт, а для нього створюємо текстові кнопки.

Жоден з ігрових об’єктів не може бути працездатним в грі без полотна. Тож в першу чергу створюємо його через натискання правої кнопки миші та обирання послідовності дій «UI» – «Panel», задаючи параметри полотна під параметри головної камери. Тепер робота можлива по всьому тлу. Розміщуємо текстові елементи (Рис. 3.2.1).

Головне меню містить такі текстові елементи:

* Назва гри «Згадай»;
* Початок гри «Нова гра»;
* Меню «Налаштування»;
* Вихід із гри «Вихід».

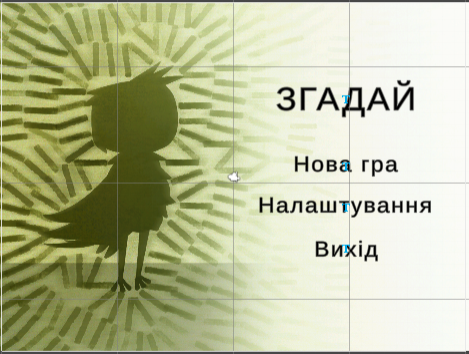


Рисунок 3.2.1 – Головне меню

У боковій панелі по праву сторону від робочого вікна редагуємо налаштування тексту (Рис. 3.2.2).

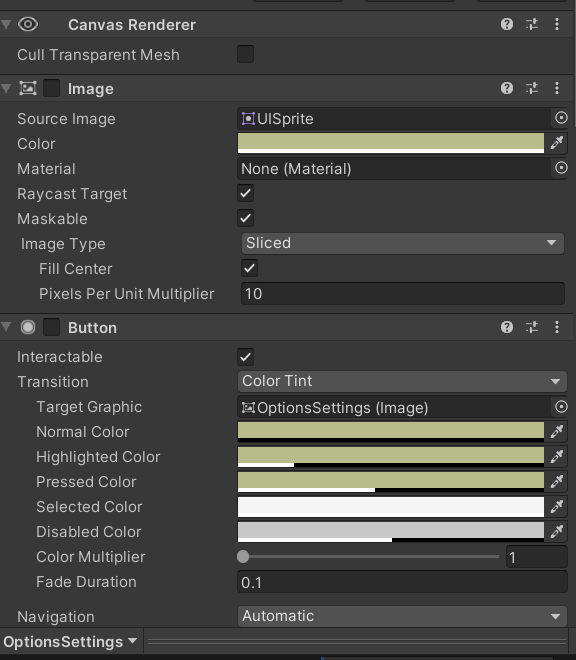


Рисунок 3.2.2 – Налаштування тексту

Меню «Налаштування» містить:

* Налаштування;
* Музика;
* Назад.

Меню розміщюється на тому тлі, що й головне меню. Для створення цього меню необхідно увесь текст головного меню згрупувати під одну UI панель та приховати його.

Щоб не повторювати знову налаштування тексту, копіюємо панель головного меню, та змінюємо назви на необхідні (Рис. 3.2.3). Єдиним додатковим елементом буде повзунок, створений через UI елемент. Він відповідатиме за гучність музики.



Рисунок 3.2.3 – Меню «Налаштування»

До налаштувань головної панелі додаємо аудіо-ресурс, у якому знаходитися фонова музика гри. Прив’язуємо повзунок із налаштувань до повзунка, що знаходиться у грі. Для цього використовуємо скрипт (Рис. 3.2.4).



Рисунок 3.2.4 – Налаштування фонового звуку

Для переключення між панеляминеобхідно в налаштуваннях кнопок («Налаштування» в MainMenu та «Назад» в OptionsMenu) в події «On Click» додати обидві панелі та через обраний скрипт «GameObject.SetActive», вбудований в рушій, залишити галочку на протилежній (Рис. 3.2.5).

За допомогою цього тепер можна заходити у меню налаштування, та повертатися до головного меню. (Рис. 3.2.6).



Рисунок 3.2.5 – Подія «On Click ()»

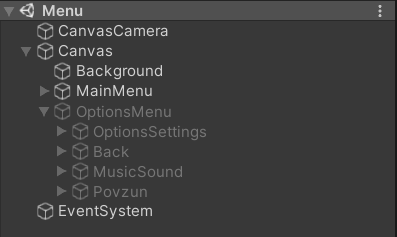


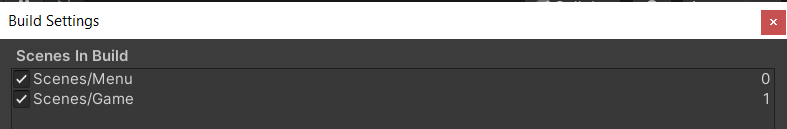
Рисунок 3.2.6 – Сховані «Налаштування»

Під час гри непотрібна панель стає невидимою.

**3.3 Розробка основної гри**

Для основної частини гри створюємо сцену «Game» за тим же принципом, що сцену «Menu». В першу чергу необхідно задати порядок сцен, щоб була можливість вільно переміщатися між ними.

У рядку меню знаходимо «File» – «Build Settings…». У відритому вікні перетягуємо наші сцени і додаємо у правильному порядку (Рис. 3.3.1).

Рисунок 3.3.1 – Налаштування порядку сцен

Перехід між сценами здійснюється за допомогою скрипту з індексом +1, який переносе гравця на сцену вище.

**Пам’ятка гравцю**

Пам’ятка створена для пояснення гравцю мети гри. Вона існує у вигляді ігрового елементу знаку питання «Question» (Рис.3.3.2). Серед компонентів в елементі додано «Button», тож це заскриптований елемент, який при натисканні розгортає заховане полотно (Рис. 3.3.3).



Рисунок 3.3.2 – Основна сцена гри

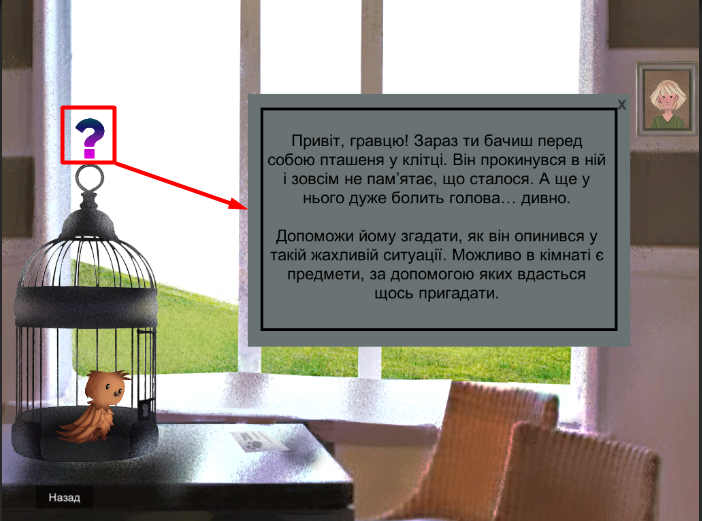


Рисунок 3.3.3 – Підказка гравцю

Елемент «Question»знаходиться в «GraphicAssets», полотно з підказкою «CanvsaD» розміщене безпосередньо під сценою, та є прямим дочірнім елементом, який при початку гри прихований.

Для відкриття та закриття «CanvsaD» у налаштуваннях «GraphicAssets» (Рис. 3.3.4) та «CanvsaD» (Рис. 3.3.5) необхідно додати обидві панелі та поставити «GameObject.SetActive» на них. На панелі «GraphicAssets» в усіх випадках буде поставлена галочка, так як при відкритті захованої панелі, інша не повинна зникати.

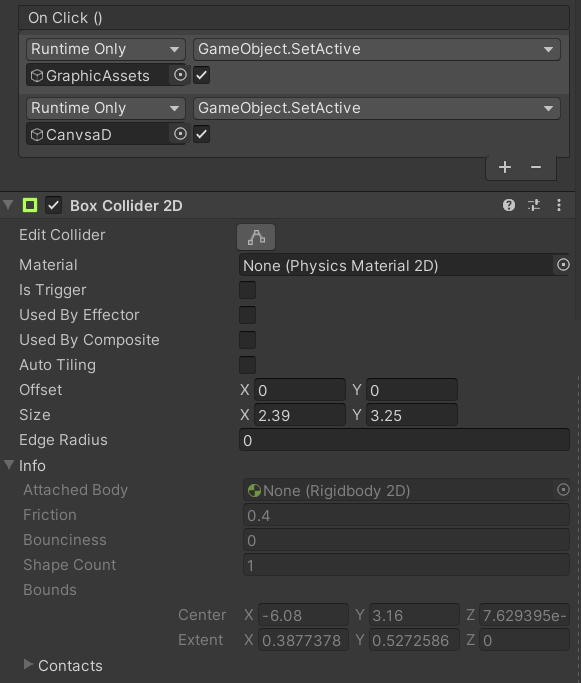


Рисунок 3.3.4 – Налаштування кнопки «Питання»

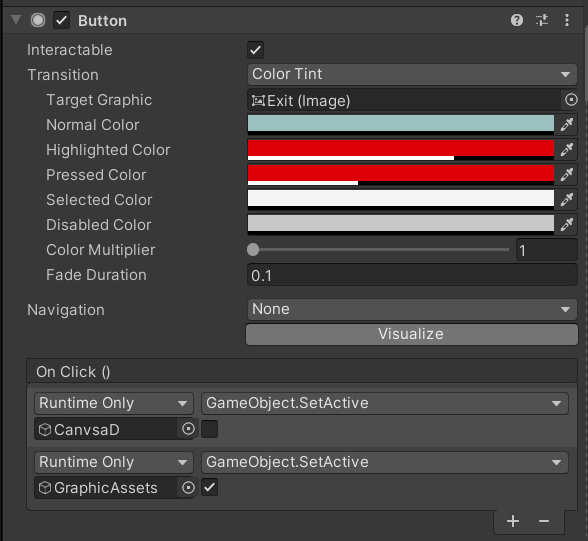


Рисунок 3.3.5 – Налаштування кнопки для закриття пам’ятки

**Кнопка повернення до головного меню**

Кнопка повернення до головного меню (Рис. 3.3.6). Повертає гравця назад у головне меню через скрипт переходу між сценами з індексом -1, який повертає гравця на сцену нижче.



Рисунок 3.3.6 – Кнопка «Назад»

**Заскриптований об’єкт**

Через нього можливо потрапити до гри Саймон. Вибір кнопки має сюжетне рішення (Рис. 3.3.7).

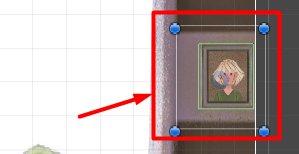


Рисунок 3.3.7 – Заскриптований об’єкт «girl»

Об’єкт перетворений на кнопку, містить у собі активну подію «On Click» для переходу на нову панель з грою (Рис. 3.3.8) він має невелику анімацію при близькому находженню миші, щоб гравцю було легше зрозуміти, що на нього треба натиснути.

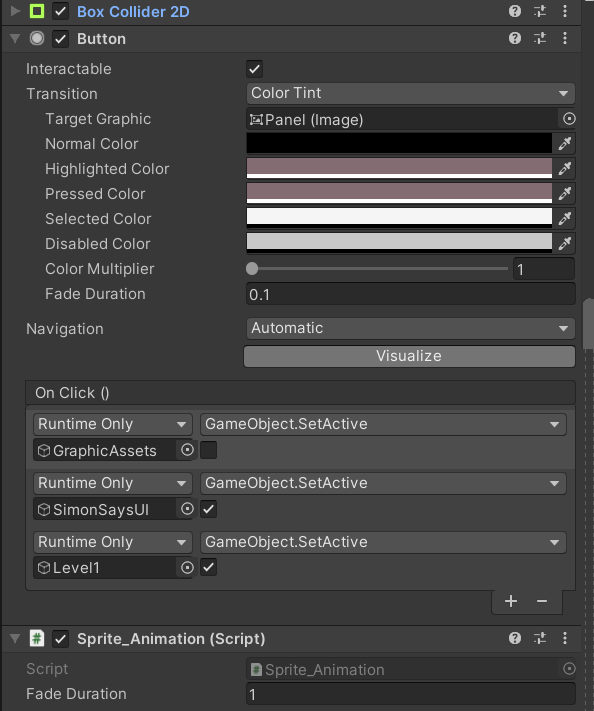


Рисунок 3.3.8 – Налаштування об’єкту

**Гра «Саймон»**

При натисканні повинна відкриватися панель з грою Саймон «SimonSaysUI» (Рис. 3.3.9).

Панель з грою містить просте тло, ігрові об’єкти спрайти різного кольору. Спрайти напівпрозорі, так як при грі вони мають підсвічуватись. Через кнопку назад можливо повернутися назад. Кнопка «Старт» починає гру.

Тут є рахунок, який визначає, скільки рівнів ти в даний момент пройшов, та найвищий рахунок. При досяганні певної відмітки, гравець буде отримувати нагороду у вигляді картинки. Після отримання усіх картинок гра закінчиться.

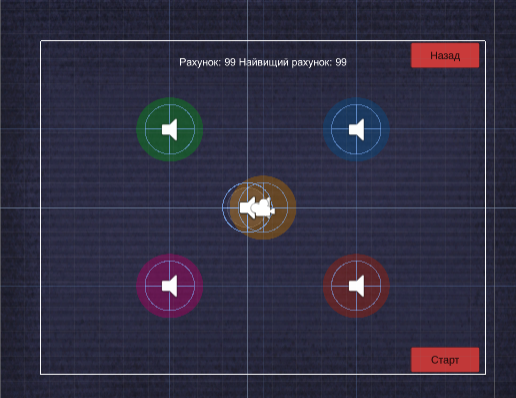


Рисунок 3.3.9 – Панель з грою «Саймон»

У налаштуваннях панелі «SimonSaysUI» через скрипт створюється публічна відкрита функція. Вона необхідна для упорядкування елементів і присвоєнню їм порядкового номеру. Аналогічно цьому створюється публічна функція для панелей, на які вже були накладені звуки. Вони будуть спрацьовувати при натисканні на кнопку. Їх порядковий номер повинен співпадати з порядковим номерів елементів «Colours» (Рис. 3.3.10).

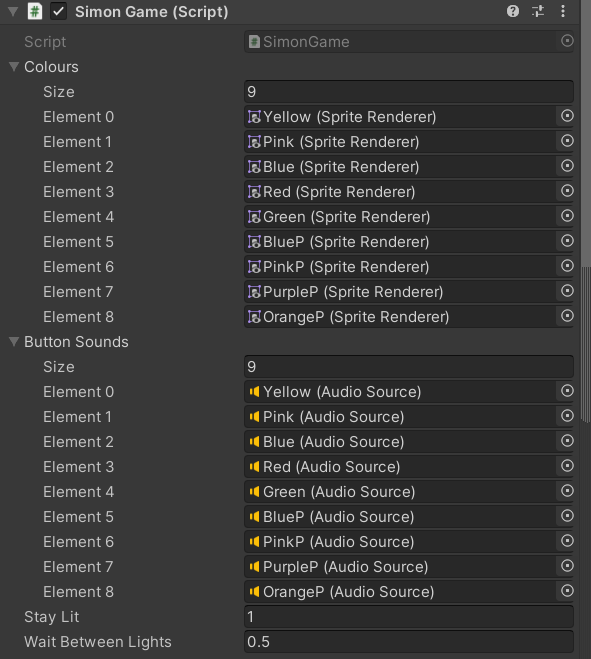


Рисунок 3.3.10 – Публічні функції для спрайтів

Якщо гравець не помиляється, в кінці кожного рівня є звук під назвою «Correct». При помилці включиться звук «Wrong», який буде позначати помилку.

Звукові елементи додаються через скрипт та розміщаються у налаштуваннях «SimonSaysUI». Тоді гра Саймон зупиняється. І гру потрібно проходити повторно.

У налаштуваннях панелі «SimonSaysUI» містяться усі елементи-нагороди (Рис. 3.3.11).

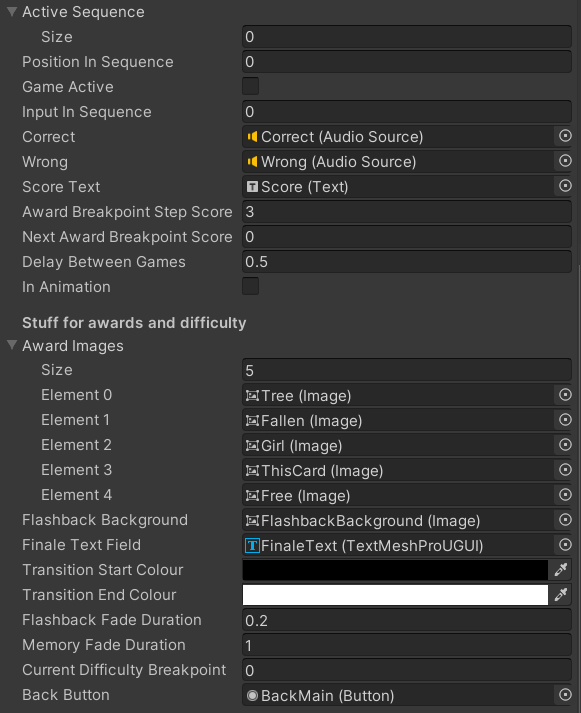


Рисунок 3.3.11 – Звукові елементи та нагороди

Програмний код реалізації компонентів наведено у Додатку А.

**РОЗДІЛ 4 ТЕСТУВАННЯ**

**4.1 Вибір виду тестування**

Тестування потрібне для перевірки програмного продукту на наявність помилок у ньому. Це необхідно для виправлення помилок до момент релізу продукту та забезпеченню максимального комфорту людині, яка користується даним продуктом.

Існує велика кількість видів тестування, яка залежить від мети гри та її об’єму. У даному випадку комп’ютерна гра невелика, та не містить велику кількість деталей. Тож для неї підходить мануальне тестування, або ж ручне. Воно дозволить пройти майбутній шлях гравця, зрозуміти наскільки вона інтуїтивно-зрозуміла.

Ціль даного тестування – перевірка відповідності результатів роботи гри до описаних на етапі проектування та розробки ТЗ.

**4.2 Тест план**

Складемо тест-плани та визначимо які функції повинні бути протестовані. За тест-планом для сцени головного меню необхідно зробити тест даних функцій:

* запуск файлу гри;
* перехід у налаштування через головне меню;
* зміна гучності музики;
* повернення з налаштувань до головного меню;
* перехід у нову гру через головне меню;
* повернення з нової гри до головного меню;
* завершення виконання файлу гри.

За тест-планом для сцени гри необхідно зробити тест даних функцій:

* відкриття підказки;
* коректне відображення підказки;
* закриття підказки;
* відкриття гри «Саймон» через ігровий об’єкт;
* повернення назад до головної сцени;
* активація гри через кнопку старт;
* правильність підрахунку пройдених рівнів;
* отримання нагороди 1-5;
* закінчення гри;
* повернення до головного меню.

Таблиця 4.1 – Тест-кейси для сцени «MainMenu»

|  |  |  |  |
| --- | --- | --- | --- |
| Опис | | Перевірка запуску файлу гри | |
| Передумови | | Гру запущено | |
| № | Дія | | Очікуваний результат |
| 1. | Запустити файл гри. | | Файл гри запускається та відображається головне меню. |
| Опис | | Перевірка меню «Налаштування» | |
| Передумови | | Знаходження у головному меню | |
| № | Дія | | Очікуваний результат |
| 1. | Перехід до меню налаштувань через кнопку «Налаштування». | | Здійснюється перехід до меню налаштувань. |
| 2. | Зміна гучності музики через повзунок. | | Гучність фонової музики змінюється. |
| 3. | Повернення до головного меню через кнопку «Назад» із збереженням рівня гучності. | | Здійснюється повернення до головного меню та зберігаються нові налаштування музики. |
| Опис | | Перевірка переходу до сцени «Game» | |

Продовження таблиці 4.1

|  |  |  |  |
| --- | --- | --- | --- |
| Опис | | Перевірка переходу до сцени «Game» | |
| Передумови | | Знаходження у сцені «MainMenu» | |
| № | Дія | | Очікуваний результат |
| 1. | Перехід до сцени «Game» через кнопку «Нова гра». | | Здійснюється перехід до сцени «Game», сцена відкривається. |
| 2. | Повернення до сцени «MainMenu» через кнопку «Назад». | | Здійснюється повернення до сцени «MainMenu». |
| Опис | | Перевірка завершення гри | |
| Передумови | | Знаходження у головному меню | |
| № | Дія | | Очікуваний результат |
| 1. | Натиснути на кнопку «Вихід». | | Завершення виконання файлу гри. |

Таблиця 4.2 – Тест-кейси для сцени «Game»

|  |  |  |  |
| --- | --- | --- | --- |
| Опис | | Перевірка роботи підказки для гравця | |
| Передумови | | Гру розпочато | |
| № | Дія | | Очікуваний результат |
| 1. | Відкриття підказки через натискання на елемент «Question». | | Відкривається полотно. |
| 2. | Перевірити наявність полотна з текстом та кнопкою закриття на ньому. | | Полотно відображається, на ньому відображається текст та кнопка закриття. |

Продовження таблиці 4.2

|  |  |  |  |
| --- | --- | --- | --- |
| 3. | Натискання на кнопку у вигляді хрестика у верхньому правому кутку на полотні. | | Закриття полотна. |
| Опис | | Перевірка можливості роботи гри «Саймон» | |
| Передумови | | Гру розпочато | |
| № | Дія | | Очікуваний результат |
| 1. | Відкриття гри «Саймон» через ігровий об’єкт «girl». | | Відкривається панель на весь екран. |
| 2. | Повернення назад через кнопку «Назад». | | Повернення до початкової панелі гри. |
| 3. | Активація гри через кнопку «Старт». | | Розпочинається гра. |
| 4. | Порівняння кількості пройдених рівнів і числу рахунку у верхньому екрані. | | Число рахунку рівнів і кількість пройдених рівнів співпадають. |
| 5. | Проходження чотирьох рівнів гри для отримання нагороди. | | Усі 5 картинок отримано після досягнення 4 рівня. |
| 6. | Перегляд після отримання нагород на наявність панелі, яка повідомляє про кінець гри. | | Після отримання нагород з’являється панель, яка повідомляє про кінець гри. |
| 7. | Перегляд повернення до головного меню після закінчення гри. | | Після закінчення гри, здійснюється повернення до головного меню. |
| № | Дія | | Очікуваний результат |
| 1. | Натиснути на кнопку «Вихід» для виходу з гри. | | Завершення виконання файлу гри. |

**ВИСНОВКИ**

Мета дипломної роботи полягала в створені гри, яка буде заохочувати гравця концентруватися на проходженні головоломок через систему нагороди, що поліпшить його увагу та зафіксує позитивний досвід навчання.

Проведена ґрунтовна робота над теоретичною частиною, з оглядом предметної області. Сформовано визначення комп’ютерних ігор, та що вони собою представляють; розглянутий їх взаємозв'язок з технологічним розвитком.

Проаналізовано тенденції, через які індустрія комп’ютерних ігор стала такою популярною та затребуваною серед населення, які фактори зіграли у цьому роль, та як змінювалося відношення до цього з часом.

Над класифікацією жанрів проведена змістовна робота. Це допомогло розумінню різниць між жанрами, їх направленості та зв’язку один з одним.

Проведення аналізу ігор допомогло визначати переваги та недоліки кожної з них, визначаючи спільні риси, притаманні жанру, та різницю, що робить кожний з проектів унікальним.

Вибір ігрового рушію проводився на основі аналізу популярних платформ для розробки ігор, після нього обраний рушій Unity, що цілком задовільнив усі потреби при створенні програми. Після цього обрано мову програмування C#.

Ігрові об’єкти гри розробленні самостійно, спеціально для даного проекту. Для їх створення використовувалися такі програми як Adobe Photoshop та DragonBones.

Розроблена двовимірна гра «Згадай» концентрує свою увагу на жанрі головоломки. В проекті представлене повноцінно працююче головне меню, в якому є можливість зайти у гру, змінити налаштування гучності звуку та здійснення виходу з гри.

Проведена робота з візуальною частиною гри для більшого інтересу молодшої аудиторії, на яку гра націлена. Окрім візуальної частини, приділена увага музикальному оформленню, для можливості глибше перейнятися до атмосфери гри.

Сама головоломка представлена у вигляді популярної для розвинення пам’яті та концентрації гри «Саймон». Для заохочення гравця проходити гру, створена система нагороди. Нагородою в грі представлені картинки, які розповідають історію. Отже, гра має розвиваючу та навчально-пізнвальну цінність.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Міні історія комп’ютерних ігор [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://publish.com.ua/istoriia/mini-istoriya-komp-yuternikh-igor.html.

2. Мисенко О. Комп’ютерна гра та авторське право [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу:

https://blog.liga.net/user/amyisenko/article/38093.

3. (Фурсова Н.А. Фактори впливу комп’ютерних ігор на розвиток суспільства / Н.А. Фурсова, О.А. Смірнова // Тези 73-ої наукової конференції професорів, викладачів, наукових працівників, аспірантів та студентів Національного університету «Полтавська політехніка імені Юрія Кондратюка (Полтава, 21 квітня – 13 травня 2021 р.). – Полтава : Національний університет імені Юрія Кондратюка, 2021. – Т. 1. – С. 468-469)

4. Video Game History Timeline [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу:

https://www.museumofplay.org/about/icheg/video-game-history/timeline.

5. Timeline of Computer History [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу:

https://www.computerhistory.org/timeline/graphics-games/.

6. Definition geek [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://whatis.techtarget.com/definition/geek.

7. Гейміфікація в освіті [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://osvitanova.com.ua/posts/2596-heimifikatsiia-v-osviti.

8. The Impact of Video Games on Culture Timeline [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://open.lib.umn.edu/mediaandculture/chapter/10-4-the-impact-of-video-games-on-culture/.

9. The Many Different Types of Video Games & Their Subgenres [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://www.idtech.com/blog/different-types-of-video-game-genres.

10. Oxford N. What's the Definition of an Action Game? [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – https://www.lifewire.com/nintendo-action-game-1126179

11. Hosch W.L. Electronic adventure game [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://www.britannica.com/topic/electronic-adventure-game

12. Bronstring M. What are adventures games? [Електронний ресурс]: [Веб-сайт]. – Електронні дані.  – Режим доступу: https://adventuregamers.com/articles/view/17547

13. Hosch W.L Role-playing video game [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://www.britannica.com/topic/role-playing-video-game

14. Marchelletta C. Explore and Play Simulation Games [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://www.lifewire.com/play-simulation-games-837139

15. Dor S. Strategy in Games or Strategy Games: Dictionary and Encyclopaedic Definitions for Game Studies [Електронний ресурс]: [Веб-сайт]. – Електронні дані.  – Режим доступу: http://gamestudies.org/1801/articles/simon\_dor

16. Puzzle video game facts for kids [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://kids.kiddle.co/Puzzle\_video\_game

17. FRACT OSC [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://vgtimes.ru/games/fract-osc/

18. The Witness (2016 video game) [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://en.wikipedia.org/wiki/The\_Witness (2016\_video\_game).

19. Crayon Physics Deluxe [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу:

https://en.wikipedia.org/wiki/Crayon\_Physics\_Deluxe.

20. What is Game Engine [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://www.igi-global.com/dictionary/game-engine/11828.

21. О.Є. Єлісеєнко Порівняльний аналіз сучасних гральних рушіїв / Єлисенко О.Є. – Вінниця: Вінницький національний технічний університет, 2016.

22. 10 найкращих ігрових рушіїв [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://ulab.sumdu.edu.ua/uk/10-najkrashhih-igrovih-rushiiv.

23. Ольховский Д. Game maker Studio – обзор программы [Електронний ресурс]: [Веб-сайт]. – Електронні ресурс. – Режим доступу: https://public-pc.com/game-maker-studio-obzor-programmyi/.

24.  Godot Engine [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://docs.godotengine.org/uk/stable/about/introduction.html.

25. Етапи розробки комп’ютерних ігор [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: https://blog.fugas.space/gamedev-stages/

26. Смірнова О.А. Інді-ігри в індустрії комп’ютерних ігор / О.А. Смірнова; наук. кер. Н.А. Фурсова // Матеріали XXVII Міжнародної науково-практичної конференції студентів, аспірантів та молодих учених «Актуальні проблеми життєдіяльності суспільства». – Кременчук : КрНУ, 2020. – С. 33

**ДОДАТОК А**  **ВИХІДНІ КОДИ ПРОГРАМНОГО ЗАСОБУ**

1. MenuS.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using DG.Tweening;

public class MenuS : MonoBehaviour

{

public void PlayGame ()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);

}

public void QuitGame ()

{

Debug.Log("Вихід");

Application.Quit();

}

private void Start()

{

DG.Tweening.DOTween.KillAll();

}

}

1. SettingsMenu.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.Audio;

public class SettingsMenu : MonoBehaviour

{

public AudioMixer audioMixer;

public void SetVolume (float volume)

{

audioMixer.SetFloat("volume", volume);

}

}

1. BackToMenu.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class BackToMenu : MonoBehaviour

{

public void BackGame()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex - 1);

}

1. ButtonClick.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ButtonClick : MonoBehaviour

{

private SpriteRenderer theSprite;

public int thisButtonNumber;

private SimonGame theSG;

private AudioSource theSound;

void OnMouseDown()

{

theSprite.color = new Color(theSprite.color.r, theSprite.color.g, theSprite.color.b, 1f);

theSound.Play();

}

void OnMouseUp()

{

theSprite.color = new Color(theSprite.color.r, theSprite.color.g, theSprite.color.b, 0.5f);

theSG.ColourPressed(thisButtonNumber);

theSound.Stop();

}

void Start()

{

theSprite = GetComponent<SpriteRenderer>();

theSG = FindObjectOfType<SimonGame>();

theSound = GetComponent<AudioSource>();

}

1. SimonGame.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using DG.Tweening;

using TMPro;

using UnityEngine.SceneManagement;

public class SimonGame : MonoBehaviour

{

public SpriteRenderer[] Colours;

public AudioSource[] buttonSounds;

private int colourSelect;

public float stayLit;

private float stayLitCounter;

private bool stayLitDone;

private bool shouldBeDark;

public float waitBetweenLights;

private float waitBetweenCounter;

public List<int> activeSequence;

public int positionInSequence;

public bool gameActive;

public int inputInSequence;

public AudioSource correct;

public AudioSource wrong;

public Text scoreText;

public int awardBreakpointStepScore = 3;

public int nextAwardBreakpointScore = 0;

// These are done with an offset of -1

protected int currentBreakpointNumber = -1;

protected int maxBreakpointNumber = 4;

protected int highestScore = 0;

protected int currentScore = 0;

public float delayBetweenGames = 0.3f;

public bool inAnimation = false;

[Header("Stuff for awards and difficulty")]

[SerializeField]

private Image[] AwardImages;

[SerializeField]

private Image \_flashbackBackground;

[SerializeField]

private TMP\_Text FinaleTextField;

[SerializeField]

private Color transitionStartColour, transitionEndColour;

[SerializeField]

float \_flashbackFadeDuration = .2f;

[SerializeField]

float \_memoryFadeDuration = 1f;

[SerializeField]

readonly int \_firstDifficultySize = 5;

[SerializeField]

readonly int \_secondDifficultySize = 7;

[SerializeField]

readonly int \_thirdDifficultySize = 9;

public int \_currentDifficultyBreakpoint = 0;

[SerializeField]

private Button \_backButton;

private int currentDifficulty = 0;

void Update()

{

if (Input.GetKeyDown(KeyCode.F))

{

HandleEndSequence();

}

if (!inAnimation)

{

if (stayLitDone)

{

stayLitCounter -= 1 \* Time.deltaTime;

if (stayLitCounter < 0)

{

Colours[activeSequence[positionInSequence]].color = new Color(Colours[activeSequence[positionInSequence]].color.r, Colours[activeSequence[positionInSequence]].color.g, Colours[activeSequence[positionInSequence]].color.b, 0.5f);

buttonSounds[activeSequence[positionInSequence]].Stop();

stayLitDone = false;

shouldBeDark = true;

waitBetweenCounter = waitBetweenLights;

positionInSequence++;

}

}

if (shouldBeDark)

{

waitBetweenCounter -= 1 \* Time.deltaTime;

if (positionInSequence >= activeSequence.Count)

{

shouldBeDark = false;

gameActive = true;

}

else

{

if (waitBetweenCounter < 0)

{

Colours[activeSequence[positionInSequence]].color = new Color(Colours[activeSequence[positionInSequence]].color.r, Colours[activeSequence[positionInSequence]].color.g, Colours[activeSequence[positionInSequence]].color.b, 1f);

buttonSounds[activeSequence[positionInSequence]].Play();

stayLitCounter = stayLit;

stayLitDone = true;

shouldBeDark = false;

}

}

}

}

}

void Start()

{

// if there's no high score, set it as 0

if (!PlayerPrefs.HasKey("HiScore"))

{

PlayerPrefs.SetInt("HiScore", 0);

}

// Assign internal value

highestScore = PlayerPrefs.GetInt("HiScore");

nextAwardBreakpointScore = nextAwardBreakpointScore + awardBreakpointStepScore;

// Set text

SetScoreInUI();

}

public void StartGame()

{

currentScore = 0;

\_currentDifficultyBreakpoint = -1;

currentDifficulty = \_firstDifficultySize;

activeSequence.Clear();

positionInSequence = 0;

inputInSequence = 0;

colourSelect = Random.Range(0, currentDifficulty);

activeSequence.Add(colourSelect);

Colours[activeSequence[positionInSequence]].color = new Color(Colours[activeSequence[positionInSequence]].color.r, Colours[activeSequence[positionInSequence]].color.g, Colours[activeSequence[positionInSequence]].color.b, 1f);

buttonSounds[activeSequence[positionInSequence]].Play();

stayLitCounter = stayLit;

stayLitDone = true;

SetScoreInUI();

}

public void ColourPressed(int whichButton)

{

if (gameActive & !inAnimation)

{

if (activeSequence[inputInSequence] == whichButton)

{

Debug.Log("Correct");

inputInSequence++;

if (inputInSequence >= activeSequence.Count)

{

// increase our score

currentScore++;

// Check for highest score

if (activeSequence.Count > highestScore)

{

highestScore++;

}

SetScoreInUI();

correct.Play();

// In case player has gathered enough, provide an award

if (currentScore % awardBreakpointStepScore == 0)

{

\_currentDifficultyBreakpoint++;

UpTheDifficulty();

}

if (currentScore < nextAwardBreakpointScore)

{

StartCoroutine(UpdateGame(delayBetweenGames));

}

else

{

Debug.Log("Awarding a memory");

inAnimation = true;

AwardMemory();

}

}

}

else

{

// Set highest score if applicable

if (highestScore > PlayerPrefs.GetInt("HiScore"))

{

PlayerPrefs.SetInt("HiScore", highestScore);

}

// Reset current score

ResetButtonsForDifficulty();

currentScore = 0;

currentDifficulty = 0;

\_currentDifficultyBreakpoint = -1;

Debug.Log("Wrong");

wrong.Play();

gameActive = false;

SetScoreInUI();

}

}

}

public void AwardMemory()

{

/\*

\* increase next breakpoint

\* raise a flag to stop a sequence for the time being

\* play animation with a target image accoring to current breakpoint

\*

\* fade animation out

\* place a frame left side

\* increase difficulty if need be

\*

\*/

currentBreakpointNumber++;

if (currentBreakpointNumber != 4)

{

nextAwardBreakpointScore += awardBreakpointStepScore;

StartCoroutine(AnimateAwardedImage(currentBreakpointNumber));

}

// else we're on a finish line

else

{

/\*

\* Show text + background

\* animate all frames one-by-one

\* animate final frame

\* animate the end

\* get to the main menu and be done with it

\*/

HandleEndSequence();

}

}

private void UpTheDifficulty()

{

Sequence s = DOTween.Sequence();

if (\_currentDifficultyBreakpoint == 0)

{

currentDifficulty = \_secondDifficultySize;

s.AppendCallback(() => Colours[5].gameObject.SetActive(true));

s.AppendCallback(() => Colours[6].gameObject.SetActive(true));

s.AppendCallback(() => Colours[5].DOFade(0.5f, 0.5f));

s.AppendCallback(() => Colours[6].DOFade(0.5f, 0.5f));

s.AppendInterval(0.5f);

}

if (\_currentDifficultyBreakpoint == 2)

{

currentDifficulty = \_thirdDifficultySize;

s.AppendCallback(() => Colours[7].gameObject.SetActive(true));

s.AppendCallback(() => Colours[8].gameObject.SetActive(true));

s.AppendCallback(() => Colours[7].DOFade(0.5f, 0.5f));

s.AppendCallback(() => Colours[8].DOFade(0.5f, 0.5f));

s.AppendInterval(0.5f);

}

}

private IEnumerator AnimateAwardedImage(int v)

{

Sequence s = DOTween.Sequence();

s.AppendCallback(() => \_flashbackBackground.DOFade(1, \_flashbackFadeDuration));

s.AppendInterval(\_flashbackFadeDuration);

s.AppendCallback(() => AwardImages[v].DOColor(transitionEndColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

s.AppendCallback(() => AwardImages[v].DOColor(transitionStartColour, \_memoryFadeDuration));

s.AppendCallback(() => \_flashbackBackground.DOFade(0, \_flashbackFadeDuration));

s.AppendInterval(\_memoryFadeDuration);

s.AppendCallback(() => inAnimation = false);

yield return new WaitForSeconds(\_flashbackFadeDuration + \_memoryFadeDuration \* 3);

StartCoroutine(UpdateGame(delayBetweenGames));

}

private void SetScoreInUI()

{

scoreText.text = string.Format("Рахунок: {0} . Найвищий рахунок: {1}", currentScore, highestScore);

}

IEnumerator UpdateGame(float \_delayBetweenGames)

{

// Wait at least some time between games

yield return new WaitForSeconds(\_delayBetweenGames);

UpdateSelectionSequence();

}

private void UpdateSelectionSequence()

{

// kind of reset game sequence

Debug.Log("Updating a sequence");

positionInSequence = 0;

inputInSequence = 0;

colourSelect = Random.Range(0, currentDifficulty);

activeSequence.Add(colourSelect);

Colours[activeSequence[positionInSequence]].color =

new Color(Colours[activeSequence[positionInSequence]].color.r, Colours[activeSequence[positionInSequence]].color.g, Colours[activeSequence[positionInSequence]].color.b, 1f);

buttonSounds[activeSequence[positionInSequence]].Play();

stayLitCounter = stayLit;

stayLitDone = true;

gameActive = false;

}

private void ResetButtonsForDifficulty()

{

Sequence s = DOTween.Sequence();

if (currentDifficulty == \_thirdDifficultySize)

{

s.AppendCallback(() => Colours[5].DOFade(0f, 0.5f));

s.AppendCallback(() => Colours[6].DOFade(0f, 0.5f));

s.AppendCallback(() => Colours[7].DOFade(0f, 0.5f));

s.AppendCallback(() => Colours[8].DOFade(0f, 0.5f));

s.AppendInterval(0.5f);

s.AppendCallback(() => Colours[5].gameObject.SetActive(false));

s.AppendCallback(() => Colours[6].gameObject.SetActive(false));

s.AppendCallback(() => Colours[7].gameObject.SetActive(false));

s.AppendCallback(() => Colours[8].gameObject.SetActive(false));

}

if (currentDifficulty == \_secondDifficultySize)

{

s.AppendCallback(() => Colours[5].DOFade(0f, 0.5f));

s.AppendCallback(() => Colours[6].DOFade(0f, 0.5f));

s.AppendInterval(0.5f);

s.AppendCallback(() => Colours[5].gameObject.SetActive(false));

s.AppendCallback(() => Colours[6].gameObject.SetActive(false));

}

}

private void HandleEndSequence()

{

SetScoreInUI();

Sequence s = DOTween.Sequence();

// okay, let's handle backbround and text

s.AppendCallback(() => \_flashbackBackground.DOFade(1, \_flashbackFadeDuration \* 2));

s.AppendInterval(\_memoryFadeDuration);

s.AppendCallback(() => FinaleTextField.DOFade(1, \_flashbackFadeDuration \* 2));

s.AppendInterval(\_memoryFadeDuration);

s.AppendCallback(() => FinaleTextField.DOFade(0, \_flashbackFadeDuration \* 2));

// I really have no desire to think about why array doesn't work, let's brute force

// And go all images fade out fade in one by one

// first image

s.AppendCallback(() => AwardImages[0].DOColor(transitionEndColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

s.AppendCallback(() => AwardImages[0].DOColor(transitionStartColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

// second image

s.AppendCallback(() => AwardImages[1].DOColor(transitionEndColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

s.AppendCallback(() => AwardImages[1].DOColor(transitionStartColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

// third image

s.AppendCallback(() => AwardImages[2].DOColor(transitionEndColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

s.AppendCallback(() => AwardImages[2].DOColor(transitionStartColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

// fourth image

s.AppendCallback(() => AwardImages[3].DOColor(transitionEndColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

s.AppendCallback(() => AwardImages[3].DOColor(transitionStartColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

// fifth image

s.AppendCallback(() => AwardImages[4].DOColor(transitionEndColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

s.AppendCallback(() => AwardImages[4].DOColor(transitionStartColour, \_memoryFadeDuration));

s.AppendInterval(\_memoryFadeDuration \* 2);

// Set End text

s.AppendCallback(() => FinaleTextField.SetText("КІНЕЦЬ"));

s.AppendCallback(() => FinaleTextField.DOFade(1, \_flashbackFadeDuration \* 2));

s.AppendInterval(3f);

s.AppendCallback(() => SceneManager.LoadScene(0));

}

}