МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп'ютерних наук та кібербезпеки

ВОЗНЯК МИКОЛА АНДРІЙОВИЧ

ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ РУШІЯ GODOT ДЛЯ РОЗРОБКИ ГРИ КОМБІНОВАНИХ ЖАНРІВ

Спеціальність: 122 Комп'ютерні науки
Освітньо-професійна програма: Комп'ютерні науки та інформаційні технології
Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр»

Науковий керівник:
Гришанович Тетяна Олександрівна,
доцент кафедри комп'ютерних наук
та кібербезпеки

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол No_____
засідання кафедри комп'ютерних наук
та кібербезпеки
від _____ 2024 р.
Завідувач кафедри
(_____) Гришанович Т. О.

Луцьк – 2024

ЗМІСТ

ВСТУП....3

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ СТВОРЕННЯ ВІДЕОІГОР....5

1.1 Комп'ютерні ігри....5

1.1.1 Історія комп'ютернних ігор....5

1.1.2 Жанри комп'ютернних ігор та їх характеристики....8

1.2 Ігрові механіки....11

1.3 Комбінування жанрів як спосіб отримання нових механік....12

1.4 Прецеденти успішного мікшування жанрів у відеоіграх....14

1.5 Ігровий рушій Godot....15

1.5.1 Поняття ігрового рушія та його функції....15

1.5.2 Ігровий рушій Godot 4 та його основна концепція....17

1.5.3 Переваги та недоліки Godot....18

1.6 Огляд аналогічних програмних розробок....20

РОЗДІЛ 2 ПРОЄКТУВАННЯ ТА РОЗРОБКА ГРИ «PIECE BY PIECE»....22

2.1 Призначення та вимоги до програмного засобу....22

2.2 Вибір моделі розробки програмного засобу....24

2.3 Загальний опис проєкту....25

2.4 Обґрунтування вибору інструментальних засобів розробки....37

2.5 Особливості програмної реалізації....39

2.6 Організація тестування та налагодження....44

2.7 Рекомендації по використанню та впровадженню програмного засобу....45

ВИСНОВКИ....46

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ....47

ДОДАТКИ....51

ВСТУП

Відеоігрова індустрія з кожним роком стає все більш популярною та прибутковою. Розробники постійно шукають нові шляхи для створення унікальних ігрових механік, здатних залучити та утримати велику кількість гравців. Однією з перспективних стратегій є мікшування різних жанрових підходів, що може призвести до народження абсолютно нових ігрових концепцій.

Мета роботи – розробити комп'ютерну гру з інноваційними геймплейними механіками шляхом поєднання елементів різних жанрів з використанням ігрового рушія Godot 4.

Для досягнення поставленої мети потрібно виконати наступні завдання:

- ознайомитися з історією розвитку комп'ютерних ігор;
- розглянути жанри комп'ютерних ігор та їх особливості;
- дослідити поняття ігрових механік, а також змішування жанрів, як один із способів їх отримання.
- проаналізувати прецеденти успішного змішування жанрів у комп'ютерних іграх;
- ознайомитись з ігровим рушієм Godot, його концепцією, функціями, перевагами та недоліками;
- здійснити огляд та аналіз комп'ютерних ігор, створених за допомогою рушія Godot;
- визначити призначення та вимоги до програмного продукту;
- обрати оптимальну модель розробки програмного продукту;
- обрати інструментальні засоби розробки;
- реалізувати ігровий програмний продукт з використанням обраних технологій;
- провести тестування та налагодження створеного програмного продукту;
- сформулювати рекомендації щодо використання та впровадження розробленої гри.

Об’єкт дослідження – процес створення інноваційних ігрових механік шляхом мікшування різних жанрових підходів.

Предмет дослідження – розробка комп’ютерної гри з поєднанням елементів пазлу, аркади, екшну та стратегії за допомогою ігрового рушія Godot

Результати роботи були представлені на XVIII Міжнародній науково-практичній конференції студентів і аспірантів «Молода наука Волині: пріоритети та перспективи досліджень» 14-15 травня 2024 року та Міжнародній науково-практичній конференції «Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем» 13-16 червня 2024 року.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ СТВОРЕННЯ ВІДЕОІГОР

1.1 Комп’ютерні ігри

1.1.1 Історія комп’ютернних ігор

Комп’ютерна гра, також відеогра – різновид гри (взаємодія за певним алгоритмом з метою розваги, навчання або тренування), процес, правила та участь у якій відбувається і обраховується за допомогою комп’ютера або іншого електронного обчислювального пристрою. [1]

Витоки комп’ютерних ігор беруть свій початок з середини XX століття, коли відбувався стрімкий розвиток електроніки та обчислювальних машин. Хоча різноманітні механічні та електронні ігрові пристрої існували ще з початку 1900-х років, проте вони не мали значного поширення.

Першими кроками до появи відеоігор стали розробки "Cathode ray tube Amusement Device" (укр. Розважальний пристрій з електронно-променевою трубкою) інженерів Томаса Голдсміта Молодшого та Істла Рея Менна у 1948 році та шахова комп’ютерна програма Алана Тюрінга у 1947 році. Спочатку ігрові програми, такі як шахи чи хрестики-нулики, створювалися в основному в рамках військових програм США з метою розробки комп’ютера, який міг би передбачати дії ворога. [2]

Перша успішна спроба створити відео-розважальний пристрій відбулася у 1958 році, коли Вільям Гіґінботам розробив "Tennis For Two" – першу інтерактивну гру, яка використовувала осцилограф в якості дисплею (рис. 1.1). Хоча сам винахідник не надавав їй великого значення і врешті-решт розібрав обладнання для використання в інших наукових проєктах, ця подія проклала шлях для подальшого розвитку відеоігор.[3]

Рисунок 1.1 - Tennis for Two на осцилографі DuMont Lab

Вже у 1960-х роках студенти Массачусетського технологічного інституту створили культову гру "Spacewar!" для мейнфрейма DEC PDP-1. Вона стала наступним кроком у розвитку індустрії та надихнула компанію Sanders Associates на ідею виробництва спеціалізованого ігрового пристрою для підключення до домашніх телевізорів.[4]

У 1972 році була випущена перша домашня ігрова консоль, Magnavox Odyssey, яка підключалася до звичайного телевізора. Висока популярність аркадної відеогри Pong (1972), а пізніше і її домашньої версії (1975), стала поштовхом до розвитку індустрії відеоігор. Комерційний успіх Pong призвів до створення великої кількості клонів від інших компаній.[5]

На фоні перенасичення ринку великою кількістю однотипних консолей, що призвело до першого обвалу індустрії відеоігор у 1978 році була випущена культова аркадна гра "Space Invaders" від японської компанії Taito (рис 1.2). Гра не тільки миттєво здобула неймовірний успіх, а й стала початком "золотої ери" аркадних відеоігор. Популярність Space Invaders надихнула багатьох виробників повернутися на ринок або ж вперше вийти на нього. Успіх гри також зумовив

поширення ігрових автоматів у громадських місцях, таких як ресторани, кінотеатри та магазини.[6]

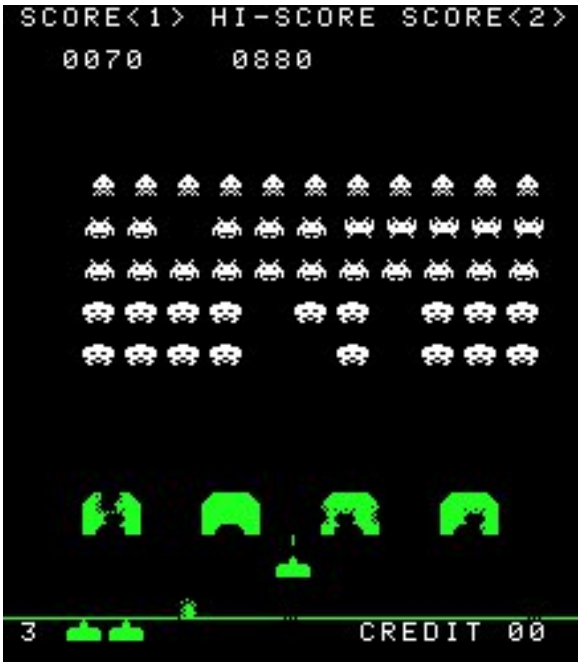


Рисунок 1.2 - гра Space Invaders

Значним поштовхом для індустрії стало впровадження CD-дисків та поява потужних персональних комп’ютерів у 1990-х. Це дозволило створювати ігри з високоякісною мультимедійною складовою, інтерактивні фільми та перші 3D-ігри.

Компанія Id Software запровадила новий підхід до створення інтерактивних продуктів, продаючи ліцензії на використання своїх ігрових рушіїв та інструментів для розробки відеоігор. Це спростило процес створення ігор і дало можливість незалежним студіям виходити на ринок. [7]

У 2000-х роках поширилися онлайн-ігри, мережеві режими, інтеграція з соціальними платформами та цифрова дистрибуція. Популярності набули модифікації від користувачів, концепції мікротранзакцій та платних доповнень. На ринок активно вийшли незалежні розробники та інді-студії.

Останніми тенденціями стали поширення 4K-графіки з фотореалістичною візуалізацією, ігри у віртуальній реальності, швидкий ріст мобільного ігрового

сегменту та прямі відео-трансляції геймплею як окремої культури у комп’ютерних іграх.

Сучасна індустрія відеоігор перетворилася на величезний багатомільярдний ринок з потужною медійною присутністю.

1.1.2 Жанри комп’ютернних ігор та їх характеристики

Жанр відеоігор – це певна категорія ігор, пов’язаних між собою схожими характеристиками ігрового процесу. Зазвичай жанри відеоігор визначаються не за сюжетом чи історією гри або її носієм, а за способом взаємодії гравця з грою.[8]

Комп’ютерні ігри охоплюють широкий спектр жанрів та піджанрів, кожен з яких має свої унікальні характеристики та особливості геймплею.

1. Екшен-ігри (Action Games) – жанр відеоігор, який вимагає від гравця швидкої реакції, координації рухів та спритності. [9] Ігри цього жанру можуть бути поділені на такі піджанри:

- Платформери (Platform games): гравець керує персонажем, який стрибає та пересувається між платформами, уникаючи перешкод та ворогів.
- Шутери (Shooter games): гравець використовує різноманітну вогнепальну зброю для знищення ворогів. існують різні типи шутерів, такі як шутери від першої особи (FPS), шутери від третьої особи (TPS) та героїчні шутери (Hero shooters).
- Файтинги (Fighting games): гравець бере участь у ближньому бою, зазвичай один на один, використовуючи різноманітні прийоми та комбінації ударів.
- Бійки (Beat ’em up): схожі на файтинги, але гравець має протистояти великій кількості ворогів одночасно.
- Стелс-ігри (Stealth games): гравець має непомітно проникати в певні місця та уникати виявлення ворогами.

2. Пригодницькі ігри (Adventure) – жанр відеоігор, зосереджений на розв’язанні головоломок, дослідженні світу та взаємодії з персонажами.[10] Можна виділити такі піджанри в цьому жанрі ігор:

- Текстові пригоди (Text Adventures): гравець вводить текстові команди для просування по сюжету.
- Графічні пригоди (Graphic adventures): гравець взаємодіє з оточенням, збираючи предмети та використовуючи їх для розв’язання головоломок.
- Візуальні новели (Visual novels): інтерактивні історії з мінімальною ігровою механікою, де гравець робить вибори, що впливають на сюжет.
- Інтерактивні фільми (Interactive movies): гравець впливає на розвиток сюжету, вибираючи дії головного героя в повноекранних відеороликах.

3. Головоломки (Puzzle) – жанр відеоігор, метою яких є вирішення логічних завдань, що вимагають від гравця задіяння логіки, стратегії, інтуїції та іноді ерудиції й уважності. [11] Серед піджанрів цього жанру можна виділити:

- Традиційна головоломка (Traditional puzzle game): зазвичай це цифрові адаптації класичних механічних чи настільних головоломок.
- Встановлення відповідності (Tile-matching game): у деяких головоломках гравцеві надаються випадкові блоки або фрагменти, які необхідно зібрати в певному порядку чи формі.
- Фізичні головоломки (Physics games): гравець має використовувати закони фізики для розв’язання головоломок.
- Пошук предметів (Hidden Object Games): гравець повинен знайти приховані на рівні предмети зі списку

4. Рольові ігри (RPGs) – жанр відеоігор, де гравець керує персонажем, розвиває його навички та просувається по сюжетній лінії. [12] Цей жанр ігор включає такі піджанри:

- Екшен з елементами рольових ігор (Action RPG): поєднують елементи ігор дії та рольових ігор, зосереджуючись на бойових системах.

- Масивні багатокористувацькі рольові онлайн-ігри (MMORPG): онлайн-ігри з великою кількістю гравців, що відіграють деякі ролі у спільному світі.
- Тактичні рольові ігри (Tactical RPG): акцентують увагу на тактичному плануванні бойових дій та розміщенні персонажів.

5. Симулятори (Simulation Games) – жанр, який намагається дати гравцю досвід перебування у наближених до реальних чи повністю вигаданих ситуаціях.[13] До цього жанру належать такі піджанри:

- Симулятори життя (Life simulation games): гравець керує віртуальними істотами, їхнім розвитком та взаємодіями.
- Симулятори транспортних засобів (Vehicle simulation games): гравець керує різноманітними транспортними засобами, такими як літаки, автомобілі чи потяги.
- Симулятори будівництва та управління (Construction and management simulation): гравець будує та керує віртуальними містами, підприємствами чи проєктами.

6. Стратегії (Strategy Games) – жанр ігор, що вимагає від гравця планування, управління ресурсами та прийняття рішень для досягнення цілей.[14] До цього жанру ігор відносяться такі піджанри:

- Глобальні стратегії (4X games): гравець має завоювати світ, досліджуючи, розширюючи, використовуючи ресурси та знищуючи ворогів.
- Стратегії реального часу (RTS): гравець керує військами в режимі реального часу, збираючи ресурси, будуючи бази та розвиваючи технології.
- Покрокові стратегії (TBS): гравець і противник ходять по черзі, плануючи свої дії та розміщуючи військові сили.
- Військові ігри (Wargames): симулюють реальні або вигадані військові конфлікти, зосереджуючись на тактиці та стратегії.
- Захист вежі (Tower defense): перед гравцем стоїть завдання захищати вежу від нападів ворогів, які наступають хвиля за хвилею, намагаючись її знищити.

7. Спортивні симулятори (Sports Games) – ігри, що імітують різні види спорту, дозволяючи гравцям брати участь у віртуальних змаганнях.[15] Ігри цього жанру поділяються на різні піджанри:

- Гоночні ігри (Racing): гравці змагаються з часом або суперниками, використовуючи різні види транспорту.
- Спортивні симулятори (Sports games): ігри, що відтворюють традиційні фізичні види спорту.
- Змагальні ігри (Competitive): мають високу конкурентну складову, але представляють вигадані чи нетрадиційні види спорту.

8. Відкриті світи та Пісочниці (Open World & Sandbox games) – жанр ігор, що, зазвичай надає гравцю повну свободу дій, творчості та довільність порядку виконання завдань. В даному жанрі можна виділити такі піджанри:

- Пісочниці-конструктори (Construction sandbox games): акцентують увагу на створенні та будівництві у відкритому світі
- Пісочниці виживання (Survival sandbox games): гравець має вижити у віртуальному світі, добуваючи ресурси, створюючи притулок і протистоячи загрозам.
- Відкриті світи пригодницького жанру (Open world adventure games): гравцю доступні локації та квести, розкидані у великих відкритих світах, часто з елементами рольових ігор.

9. Масові багатоосібні онлайн-ігри (MMO games) – жанр ігор, де велика кількість гравців взаємодіє у спільному онлайн-світі.

1.2 Ігрові механіки

Ігрові механіки є фундаментальними компонентами будь-якої відеогри, адже вони визначають правила взаємодії гравця з ігровим світом та формують основу ігрового процесу. Вони представляють собою набір можливих дій, які гравець може виконувати в рамках гри, а також систему реакцій та наслідків на ці дії.

Не існує загальноприйнятого визначення терміну "ігрова механіка", проте є кілька підходів до його тлумачення. Одні визначення розглядають ігрові механіки як "системи взаємодії між гравцем і грою", інші – як "правила та процедури, що керують діями гравця та відповідями гри на ці дії". Деякі визначення об’єднують ці два аспекти, визначаючи ігрові механіки як "будь-яку частину системи правил гри, що охоплює один і тільки один спосіб інтерактивної взаємодії, що відбувається під час гри". [16]

Для опису ігрового процесу розрізняють поняття головної ігрової механіки (ядро геймплею), основних та другорядних механік. Головна ігрова механіка описує дії, які гравець виконує знову і знову протягом усієї гри (наприклад, стрибки в платформері). Основні механіки включають усі механіки, доступні гравцю протягом усієї гри або ті, що стають доступними на ранніх стадіях гри (стрілянина, поєдинки в ближньому бою, водіння у Grand Theft Auto). Другорядні механіки можуть застосовуватися гравцем за певних умов, наприклад, після досягнення ігровим персонажем певного рівня майстерності.[17]

Ігрові механіки в основному тісно пов’язані з жанром гри, оскільки кожен жанр має свої характерні ігрові цикли та типові сценарії для гравця, однак вони не є чимось статичним та незмінним. Навіть у рамках одного жанру існує безліч варіацій механік, а інноваційні розробники постійно шукають нові підходи до їх вдосконалення та оновлення. Створення свіжих, незвичайних ігрових механік є одним із ключових факторів, що допомагає відеогрі вирізнятися на тлі інших представників свого жанру.

1.3 Комбінування жанрів як спосіб отримання нових механік

Одним з найпопулярніших і найефективніших способів створення інноваційного ігрового процесу та нового користувацького досвіду є поєднання ігрових механік з різних жанрів. Мікшування елементів кількох жанрів дозволяє досягти свіжого, незвичного геймплею, який вигідно відрізняється від типових

жанрових формул і якого не можна отримати, розробляючи гру в рамках лише одного жанру.

Головний секрет успішного жанрового міксування полягає не в тому, щоб просто з'єднати дві окремі гри в одній, а в тому, щоб створити органічне поєднання, де елементи різних жанрів гармонійно переплітаються в єдиному ігровому циклі. Для цього розробники повинні знайти центральну ігрову петлю, яка інкорпорує механіки з декількох жанрів, замість того, щоб постійно переключатися між двома різними ігровими циклами. [18]

Комбінування жанрів відкриває нові можливості для залучення різних груп гравців та розширення цільової аудиторії. Наприклад, додавання RPG-елементів, таких як захоплююча сюжетна лінія чи глибока система розвитку персонажів, може зацікавити гравців, які зазвичай ігнорують певний жанр. Це дозволяє охопити нові демографічні групи та задовольнити різні ігрові вподобання.

Крім того, жанровий гібрид може зустріти менше конкуренції на ринку, оскільки він представляє відносно нову нішу. У такій ситуації розробники можуть скористатися перевагами першопрохідців та зайняти вигідну позицію до того, як ринок розігріється.

Однак слід враховувати й певні виклики, пов'язані з комбінуванням жанрів. Зокрема, підтримувати баланс між різними елементами (художній стиль, рівень складності тощо) може бути складно, коли цільова аудиторія гри не є чітко визначеною. [19]

Загалом мікшування жанрів є потужним інструментом для створення нових ігрових механік та надання гравцям незвичайного користувацького досвіду. Хоч цей підхід і має свої виклики та ризики, але він стимулює креативність, сприяє оновленню індустрії відеоігор та відкриває нові можливості для залучення гравців.

1.4 Прецеденти успішного мікшування жанрів у відеоіграх

Хоча поєднання різних жанрів у відеоіграх є складним завданням, історія індустрії демонструє чимало прикладів успішного впровадження такого підходу.

Одним із найяскравіших зразків успішного жанрового мікшування є серія ігор Persona від Atlus. У цих іграх гармонійно поєднуються візуальна новела та пригодницька рольова гра з тактичними боями. Поряд із захоплюючим сюжетом і розвитком персонажів, гравці можуть насолоджуватися глибокою бойовою системою, заснованою на використанні "персон" – утілень духів і демонів. Persona вдалося створити незабутній ігровий досвід, який припав до смаку мільйонам гравців по всьому світу.

Іншим прикладом є культова гра Rune Factory, яка поєднує елементи фермерського симулятора з пригодницькою рольовою грою та бойовими механіками в стилі hack-and-slash. Гравці можуть розвивати своє господарство, вирощувати врожаї, одружуватися та заводити сім’ю, а також вирушати на пригоди, досліджувати підземелля та билися з монстрами. Цей унікальний мікс жанрів знайшов свою аудиторію та сформував лояльну фанатську базу.

Не можна оминути увагою й гру Cult of the Lamb від Massive Monster. Вона поєднує елементи roguelike, бойових арен і містобудування з управлінням культом. Гравці повинні керувати селищем своїх прихильників, забезпечувати їх благополуччя, а також вирушати в жорстокі підземелля, щоб боротися з ворогами. Незважаючи на деякі огріхи в узгодженні жанрових елементів, Cult of the Lamb отримала високі оцінки критиків і стала популярною завдяки своєму унікальному та чарівному міксу механік.

Крім того, варто згадати й всесвітньо відому серію Fallout від Bethesda. Ігри цієї серії успішно поєднують постапокаліптичний відкритий світ з елементами рольової гри, пригодницького бойовика та містобудування. Особливо це стосується Fallout 4, де гравці можуть не лише досліджувати світ, битися з ворогами та виконувати квести, а й будувати та розвивати своє поселення.

Отже, хоча поєднання жанрів є складним викликом, існує чимало прикладів ігор, які досягли успіху завдяки мікшуванню різних ігрових елементів. Головною запорукою успіху є гармонійне та злагоджене поєднання цих елементів у єдиний, цілісний ігровий досвід, який би був захопливим і привабливим для широкої аудиторії гравців. Вдалі прецеденти демонструють, що міксування жанрів, при належному виконанні, може стати джерелом інновацій та креативності в індустрії відеоігор.

1.5 Ігровий рушій Godot

1.5.1 Поняття ігрового рушія та його функції

Ігровий рушій (англ. Game engine) – це центральна програмна частина будь-якої відеогри, що відповідає за всю її технічну сторону. Він спрощує розробку гри шляхом уніфікації та систематизації її внутрішньої структури. Важливою перевагою рушіїв є можливість створення багатоплатформових ігор, які можуть одночасно запускатися на різних пристроях, включаючи персональні комп’ютери, ігрові консолі та мобільні пристрої. [20]

Ігровий рушій є центральним програмним компонентом будь-якої відеогри, який відповідає за всю її технічну складову. Він дозволяє полегшити процес розробки гри шляхом уніфікації та систематизації її внутрішньої структури. Важливою перевагою рушія є можливість створення багатоплатформових ігор, які можуть одночасно запускатися на різних пристроях, включаючи персональні комп’ютери, ігрові консолі та мобільні пристрої.

Основну функціональність гри забезпечує її рушій, до складу якого входять різноманітні підсистеми та компоненти [21]:

- Рушій рендерингу, або візуалізатор, відповідає за відображення графіки та візуальних ефектів.
- Фізичний рушій керує імітацією фізичних законів у віртуальному середовищі, таких як гравітація, зіткнення та взаємодії.

- Система звуку обробляє аудіоефекти та музичний супровід.
- Система скриптів дозволяє розробникам програмувати поведінку ігрових об’єктів і правила гри.
- Анімаційна система відповідає за плавний рух і перетворення ігрових персонажів та об’єктів.
- Система ігрового штучного інтелекту керує поведінкою неігрових персонажів та інших елементів гри, що потребують певної логіки прийняття рішень.
- Мережевий код забезпечує можливість багатокористувацької гри та синхронізацію даних між клієнтами.
- Система керування пам’яттю ефективно розподіляє та звільняє ресурси оперативної пам’яті під час роботи гри.
- Багатопотоковість дозволяє розпаралелювати обчислення для підвищення продуктивності.
- Граф сцени представляє ієрархічну структуру ігрового світу, спрощуючи дизайн рівнів і рендеринг складних віртуальних середовищ.

Використання ігрового рушія дозволяє заощадити час і ресурси під час розробки, оскільки основні технічні компоненти вже реалізовані та оптимізовані. Це дає змогу розробникам зосередитися на створенні ігрового контенту, такого як графіка, анімації, звукові ефекти та логіка гри. Крім того, ігрові рушії зазвичай є багаторазово використовуваними, що дозволяє створювати декілька різних ігор на основі одного рушія, змінюючи лише ігровий вміст.

На ринку існує низка популярних ігрових рушіїв, серед яких варто згадати Unity, Unreal Engine та Godot.

Unity є одним із найпоширеніших і найпотужніших рушіїв, який підтримує розробку ігор для численних платформ, включаючи ПК, консолі, мобільні пристрої та веббраузери. Він має зручне інтегроване середовище розробки, потужні інструменти для створення ігрового вмісту та підтримує кросплатформну розробку завдяки використанню мови програмування C#. [22]

Unreal Engine від Epic Games є ще одним провідним ігровим рушієм, який використовується для створення високоякісних 3D-ігор та інтерактивних застосунків. Він відомий своїми передовими можливостями рендерингу, фізичним рушієм і підтримкою реалістичної графіки. Unreal Engine також має потужний редактор рівнів та інструменти для створення ігрового вмісту. [23]

Godot – це вільний і відкритий ігровий рушій, який набуває популярності завдяки своїй простоті використання та багатофункціональності. Він підтримує розробку 2D та 3D ігор для різних платформ, включаючи ПК, мобільні пристрої та веббраузери. Godot має інтуїтивний інтерфейс, зручні інструменти для створення ігрового вмісту та підтримує кілька мов програмування, таких як GDScript, C# та C++. [24]

Варто зазначити, що крім згаданих рушіїв, існує багато інших, як комерційних, так і відкритих, кожен з яких має свої особливості та переваги залежно від вимог проєкту та уподобань розробників. Вибір рушія часто залежить від масштабу проєкту, доступних ресурсів, необхідної функціональності та цільових платформ. Незалежно від обраного рушія, він відіграє ключову роль у розробці відеоігор, забезпечуючи технічну основу та інструменти для створення захопливих ігрових світів і неперевершеного ігрового досвіду.

1.5.2 Ігровий рушій Godot 4 та його основна концепція

Godot 4 – це сучасний і потужний рушій з відкритим вихідним кодом, орієнтований на створення ігор та інтерактивних додатків. Його основною концепцією є забезпечення зручного та гнучкого інструменту для розробників, який би дозволяв ефективно реалізовувати ідеї, незалежно від масштабу проєкту чи наявного досвіду.

Фундаментальною філософією Godot є надання розробникам максимального контролю та свободи у створенні їхніх проєктів. На відміну від

деяких комерційних рушіїв, Godot не нав’язує певні практики чи обмеження, а натомість надає розробнику повну творчу свободу.

Центральним елементом Godot є система вузлів (nodes), які являють собою ієрархічні блоки, з яких формується структура гри чи додатку. Кожен вузол може містити різноманітні компоненти, такі як спрайти, моделі, фізичні тіла, скрипти, анімації тощо. Така архітектура забезпечує зручну композицію складних сцен з можливістю повторного використання окремих елементів, сприяючи модульності та гнучкості розробки.

Godot прагне забезпечити інтуїтивний та доступний підхід до створення ігор, не жертвуючи при цьому потужністю та функціональністю. Рушій пропонує власну скриптову мову GDScript, яка є легкою у вивченні, особливо для новачків, проте водночас досить гнучкою та потужною для складних проєктів. Проте Godot також підтримує C# та інші мови програмування завдяки відкритій та розширюваній структурі, що дозволяє залучати сторонні бібліотеки та інструменти.

У Godot 4 представлено потужні інструменти для розробки як 2D, так і 3D проєктів в єдиному інтегрованому середовищі. Для 2D ігор рушій пропонує зручну систему тайлмапів та спрайтів, тоді як для 3D доступний повний набір можливостей, включаючи імпорт 3D моделей, анімацій, систем освітлення та фізичних симуляцій. Ця універсальність дозволяє розробникам легко переходити між 2D та 3D проєктами, не змінюючи середовище розробки. [25]

Загалом Godot 4 втілює філософію відкритості, гнучкості та зручності, що робить його привабливим вибором для широкого кола розробників, від невеликих команд до великих студій та навіть освітніх проєктів. Орієнтація на свободу та контроль над процесом розробки, поєднана з потужними інструментами та функціями, робить Godot унікальним явищем у світі ігрових рушіїв.

1.5.3 Переваги та недоліки Godot

Рушій Godot має ряд переваг, серед яких:

- Godot має повністю відкритий вихідний код, що дозволяє будь-кому допомагати розробникам модифікувати та розширювати функціональність рушія, враховуючи потреби аудиторії.
- проєкти, створені в Godot, можуть бути легко розгорнуті на різних платформах, включаючи Windows, macOS, Linux, Android, iOS, веб-браузери та консолі, без значних змін у коді. Це дозволяє розробникам охоплювати широку аудиторію з одного проєкту.
- Godot був розроблений з акцентом на продуктивність, що забезпечує ефективне використання ресурсів і високу швидкість роботи, навіть на мобільних пристроях.
- Godot є повністю безкоштовним і не вимагає сплати роялті чи ліцензійних зборів, що робить його привабливим вибором для незалежних розробників та малих студій.
- Godot має мультимовну підтримку. Окрім вбудованої скриптової мови GDScript, Godot підтримує C#, а також дозволяє інтегрувати інші мови програмування завдяки відкритій архітектурі.
- Godot пропонує комплексний набір інструментів для розробки як 2D, так і 3D ігор та додатків в єдиному середовищі, включаючи систему тайлмапів, імпорт 3D моделей, анімацій, систем освітлення та фізичних симуляцій.

Однак Godot має і ряд недоліків:

- У порівнянні з комерційними рушіями, такими як Unity чи Unreal Engine, Godot має меншу базу користувачів та ресурсів, що може ускладнити пошук допомоги чи навчальних матеріалів.
- Оскільки Godot є відкритим проєктом, деякі більш специфічні функції можуть бути не повністю реалізовані або вимагати додаткових зусиль для інтеграції.

- Як відкритий рушій, Godot може зіткнутися з проблемами сумісності з певними зовнішніми бібліотеками чи інструментами, якщо вони не були спеціально розроблені для Godot.
- Хоча Godot має потужну систему вузлів та сцен, він не пропонує такої ж гнучкості у візуальному програмуванні, як деякі комерційні рушії.
- У порівнянні з більш популярними рушіями, Godot має менший вибір готових асетів, плагінів та розширень, доступних для придбання або безкоштовного використання.

Незважаючи на ці недоліки, Godot залишається потужним і гнучким інструментом, особливо для незалежних розробників, малих студій та освітніх проєктів. Його відкритість, безкоштовність, кросплатформеність та активна спільнота розробників роблять його привабливим вибором для тих, хто цінує свободу та контроль над своїми проєктами.

1.6 Огляд аналогічних програмних розробок

"A Most Extraordinary Gnome" (https://store.steampowered.com/app/2089710/A_Most_Extraordinary_Gnome/) – коротка пригодницька гра від Save Sloth Studios, що є чудовим прикладом можливостей рушія Godot. Гра поєднує унікальну 2D-дію з складним оповіданням, має глибоко продуманий сюжет та битви з босами. Особливостями гри є також 2D-середовище, зроблене в стилі паперових вирізок та запам’ятовувана музика у східноєвропейському стилі (рис. 1.4). [26]

Рисунок 1.3 - Знімок екрану гри "A Most Extraordinary Gnome"

"Brotato" (https://store.steampowered.com/app/1942280/Brotato/) – рогалик-шутер від Blobfish, що є ще однією чудовою ілюстрацією можливостей рушія Godot. Гра має багато функціональних особливостей, таких як автоматичний вогонь зі зброї за замовчуванням, можливість ручного прицілювання, десятки персонажів для налаштування забігів, сотні предметів та видів зброї на вибір, виживання у хвилях ворогів від 20 до 90 секунд, збір матеріалів для отримання досвіду та покупки предметів між хвилями, а також налаштування доступності: зміна здоров'я, шкоди та швидкості ворогів (рис. 1.4). [27]

Рисунок 1.4 - Знімок екрану гри "Brotato"

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ГРИ «PIECE BY PIECE»

2.1 Призначення та вимоги до програмного засобу

Програмний засіб "Piece by Piece" є відеогрою жанру пазл-аркада з елементами екшену та стратегії. Основним призначенням гри є демонстрація ігрового процесу, основою якого є змішування різних ігрових жанрів та надання користувачу можливості отримання відповідного ігрового досвіду.

Програмний засіб повинен відповідати наступним функціональним вимогам:

1. Реалізоване головне меню з наступним вмістом:
   - Кнопка "Гра" для переходу до меню рівнів.
   - Кнопка "Налаштування" для відкриття меню налаштувань.
   - Кнопка "Мануал" для перегляду інструкцій.
   - Кнопка "Вихід" для виходу з гри.
2. Реалізоване меню рівнів з наступним вмістом:
   - Кнопки для вибору та запуску окремих рівнів.
   - Відображення статусу рівнів (розблоковані, заблоковані, пройдені).
3. Реалізована ігрова логіки та механік рівня, серед яких:
   - Поява гравця на ігровій площині.
   - Перманентний рух гравця та можливість зміни напряму руху як за допомогою клавішного набору “WASD” так і за допомогою клавіатурних стрілок.
   - Систематична поява на полі фрагментів принаймні двох різних типів, з якими має взаємодіяти гравець.
   - Коректна взаємодія гравця зі звичайними фрагментами, а саме – приєднання фрагментів до структури гравця в місці, де відбулося зіткнення.
   - Коректна взаємодія гравця із вибуховим фрагментами, а саме – знищення одного фрагменту структури гравця, з яким відбулася колізія.

- Реалізація краю ігрового поля та реакції гравця на вихід за нього, а саме – знищення ряду фрагментів зі сторони, що вийшла за межі поля.
- Реалізація прикладу фінальної структури, якої повинен досягнути гравець.
- Реалізація проходження ігрового рівня у випадку, якщо фігури гравця та прикладу співпадають.
- Реалізація непроходження ігрового рівня, у випадку, якщо структура гравця не містить жодного фрагменту.
- Можливість поставити гру на пазу за допомогою клавіші “ESC” або кнопки на екрані.
- Можливість відновити ігровий процес без втрат у перебігу ігрових подій.

4. Екран налаштувань з мінімальними налаштуваннями гри.
5. Екран мануалу з базовою інформацію про геймплей.
6. Екран виграшу, що з’являється у випадку успішного проходження рівня та має наступний вміст:
   - Відомості про проходження рівня та статистика.
   - Кнопка "Знову" для перепроходження рівня.
   - Кнопка "У меню", яка відкриває меню рівнів.
7. Екран програшу, що з’являється у випадку непроходження рівня та має наступний вміст:
   - Кнопка "Знову" для перепроходження рівня.
   - Кнопка "У меню", яка відкриває меню рівнів.
8. Екран паузи, що з’являється у відповідь на призупинення гри та має наступний вміст:
   - Кнопка "Знову" для перепроходження рівня.
   - Кнопка "У меню", яка відкриває меню рівнів.
   - Кнопка "Налаштування" для відкриття меню налаштувань.
   - Кнопка "Мануал" для перегляду інструкцій.

2.2 Вибір моделі розробки програмного засобу

Для реалізації відеогри було обрано ітераційну модель розробки, яка є різновидом гнучкої методології Agile і ефективно застосовується для проєктів із змінними вимогами та високим ступенем невизначеності. Ітераційна модель передбачає циклічний підхід до створення програмного продукту, розбиваючи весь процес на послідовні ітерації або спринти.

Кожна ітерація є замкнутим циклом, що охоплює весь процес розробки програмного забезпечення: аналіз вимог, дизайн, проєктування, створення візуалу, праця в редакторі, кодування і зрештою тестування (рис. 2.1). На першій ітерації реалізується базовий функціонал, а в подальших циклах вимоги переглядаються, існуючі можливості удосконалюються, усуваються помилки та додаються нові функції.

Рисунок 2.1 - Цикл ітераційної моделі розробки

Ітераційний підхід має низку суттєвих переваг:

- Гнучкість та здатність швидко реагувати на зміни вимог на будь-якій стадії розробки, адаптуючи функціонал системи;
- Раннє отримання базової робочої версії програми та можливість її тестування користувачами для збору відгуків;
- Регулярний огляд проміжних результатів, контроль якості та пріоритезація завдань на кожній ітерації;
- Зменшення ризиків невизначеності за рахунок розбиття процесу на короткі цикли та можливості гнучкого коригування напрямку розробки;
- Оптимізація розподілу ресурсів та зосередження зусиль на найбільш критичних елементах у межах ітерації.

Застосування ітераційної моделі для створення відеогри є виправданим рішенням, адже дозволяє поетапно реалізовувати та тестувати ключові ігрові механіки, гнучко реагувати на відгуки користувачів, коригувати пріоритети та вносити необхідні зміни на кожній ітерації. Це забезпечує формування якісного ігрового досвіду, що відповідатиме очікуванням гравців.

2.3 Загальний опис проєкту

Увесь процес розробки програмного забезпечення було розділено на підзадачі, після виконання яких програмний засіб отримував новий, орієнтовно доконаний функціонал.

Спочатку було розроблено основну концепцію та механіки гри.

Основною метою гравця є формування певної структури на ігровому полі відповідно до наданого зразка. Основні елементи гри – ігрове поле, зразок, структура гравця, фрагменти, край ігрового поля. (рис. 2.2)

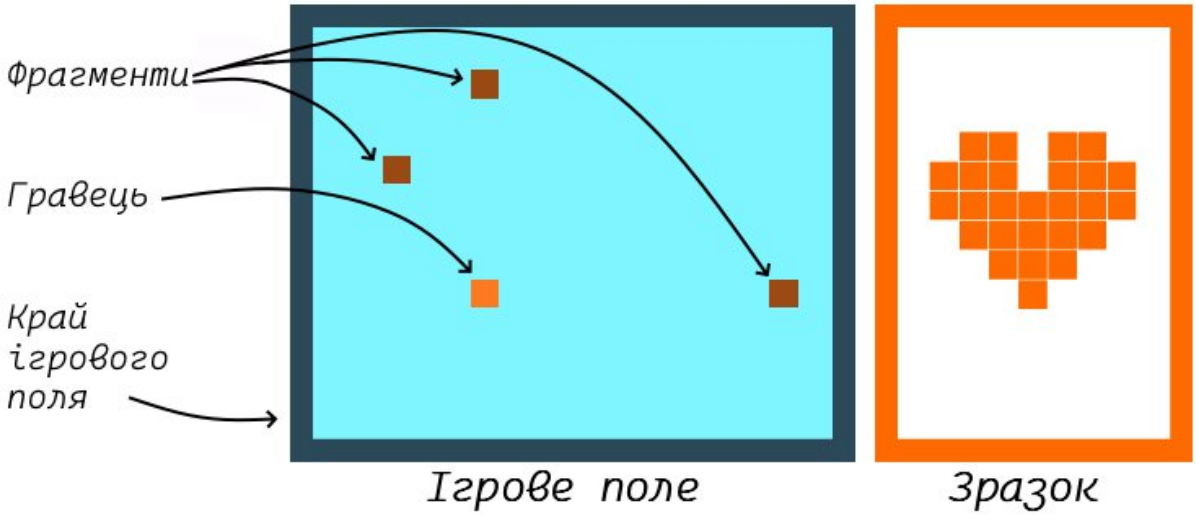


Рисунок 2.2 - Концепт основних ігрових елементів

На початку гри гравець являє собою один із фрагментів кінцевої фігури з постійним рухом, напрям якого можна змінювати клавішами керування. Безпосередньо під час гри гравець повинен приєднувати до себе інші фрагменти, що періодично з’являються на ігровому полі, доторкаючись до них. Приєднані фрагменти стають частиною структури гравця. Приєднуючи правильні фрагменти у правильних місцях, структура гравця в решті решт набуде ідентичного до зразка вигляду, що є основною ціллю гри, а отже, успішним її завершенням. (рис. 2.3) У випадку побудови хибної конструкції, гравець може видалити помилкові фрагменти, вийшовши за ігрове поле, однак така дія видалить одразу весь ряд фрагментів, що опинилися поза ігровим полем. Відповідно, якщо таким чином знищити всі фрагменти структури гравця, тобто самого гравця, гра закінчиться поразкою.

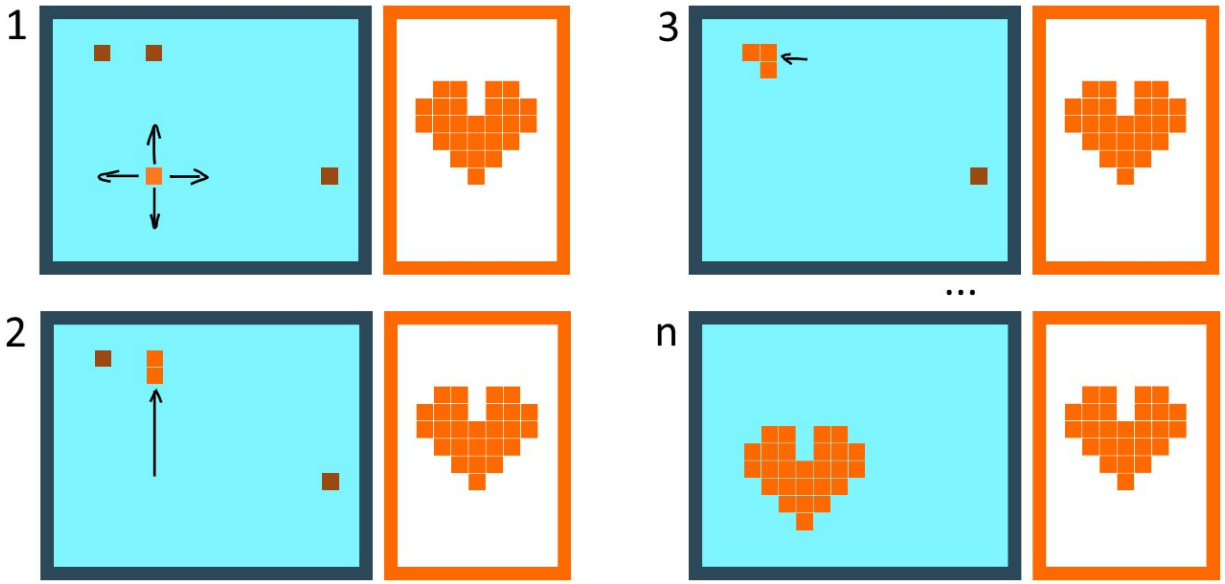


Рисунок 2.3 - Хід гри (ранній концепт)

Концепція гри відкрита для масштабування – можна додати фрагменти, що будуть виступати в ролі перешкод чи, навпаки, в ролі деяких бонусів, що матимуть різний вплив на гравця.

У цій грі було поєднано наступні жанри [28]:

- Пазл (Puzzle) – формування кінцевої структури за заданим зразком шляхом поступового приєднання фрагментів нагадує процес складання пазлів.
- Аркада (Arcade) – обмеженість ігрового простору одним полем та циклічність базових ігрових дій є характерними рисами аркадного жанру, присутніми в цій концепції.
- Екшн (Action) – потреба швидко реагувати, змінювати напрямок руху для уникнення загроз та руйнувань вимагає від гравця певних екшн-навичок.
- Стратегія (Strategy) – необхідність планувати рухи наперед та обдумувати послідовність приєднання фрагментів для побудови необхідної конструкції додає стратегічну складову.

Наступним кроком стало продумування сеттінгу гри. Оскільки концепція гри полягає у збиранні певної конструкції з окремих фрагментів, що нагадує процес реставрації, першою ідеєю став образ археолога в пустелі, який

відновлює стародавню вазу з уламків, відкопаних під час розкопок. Відштовхуючись від цієї ідеї, ігровий процес було вирішено розмістити в умовах піщаної місцевості, де гравець, виступаючи в ролі археолога, повинен зібрати вазу на робочому столі, дотримуючись зразка, зображеного на древньому сувої. Окремі фрагменти вази має підкидати на ігрове поле рука невідомого археолога, створюючи ефект знахідки під час розкопок.

Чітке бачення основного сеттингу гри дозволило приступити до наступного кроку – розробки та створення необхідного графічного оформлення та елементів.

Під час розробки концепції фрагментів стало зрозуміло, що прості квадратні фрагменти не зможуть достатньо точно передати форми складних фігур, які мали б з них збиратися. Було прийнято рішення зробити фрагменти різноманітних форм, що дозволить більш точно відтворювати контури збираних об’єктів. Щоб віднайти потрібні форми було проведено аналіз принципів мозаїчних композицій, графічних одиниць побудови дизайну на деяких сайтах конструкторах (рис. 2.4) та різних графічник стилів.

В основу дизайну нових фрагментів були покладені елементи відомого художнього стилю Баухауз (Bauhaus). Цей вибір невипадковий, адже Баухауз відомий своїми лаконічними, геометричними та графічними формами. (рис. 2.5) Саме такий мінімалістичний та водночас виразний дизайн ідеально підходив для створення фрагментів, які б мали складати цілісні образи об’єктів у грі. Геометричність форм Баухаузу дозволяла легко комбінувати фрагменти різних обрисів у складні структури.

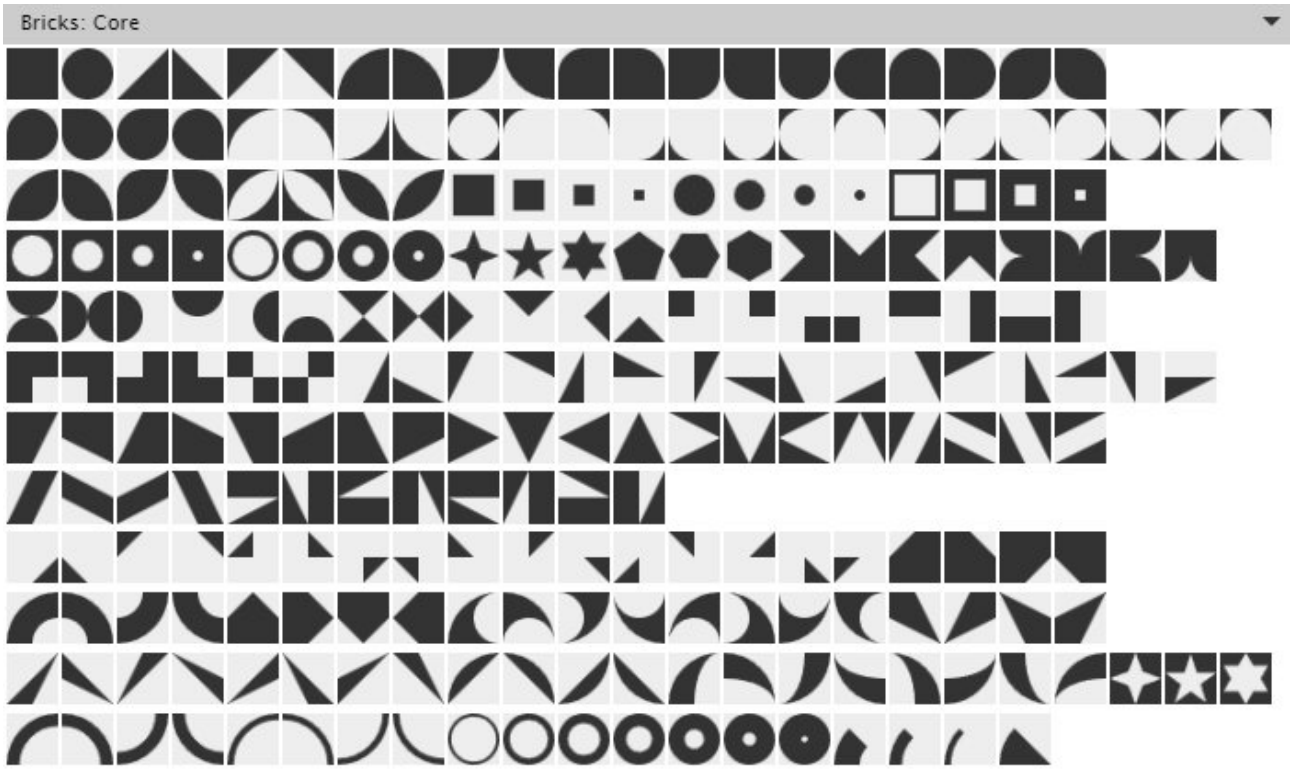


Рисунок 2.4 - Набір графічних фігур для побудови шрифтів з сайту fontstruct.com

Рисунок 2.5 - Приклади робіт у стилі Bauhaus

Для візуального оформлення було обрано піксельну графіку. Цей вибір зумовлений кількома факторами. По-перше, піксельний стиль додавав особливої ретро-атмосфери та шарму, відсилаючи до ранніх відеоігрових епох.

Крім того, використання піксельної графіки низького розширення дозволило уникнути низки візуальних проблем. При піксельній графіці площина

дотику між двома елементами, наприклад, трикутними фрагментами, є відносно більшою, ніж при високороздільному зображенні. Припустимо, два трикутні фрагменти з’єднуються в одній восьмій своєї площини при низькому розширенні. В той час як при високій роздільній здатності площина з’єднання може становити лише одну двохсотту частину. Це створює зовсім відмінне візуальне сприйняття з’єднання та цілісності побудованої структури.

В результаті було сформовано набір фрагментів, що мали зрізані краї, незвичайні геометричні форми, дугоподібні вигини та навіть візерунки (рис. 2.6).

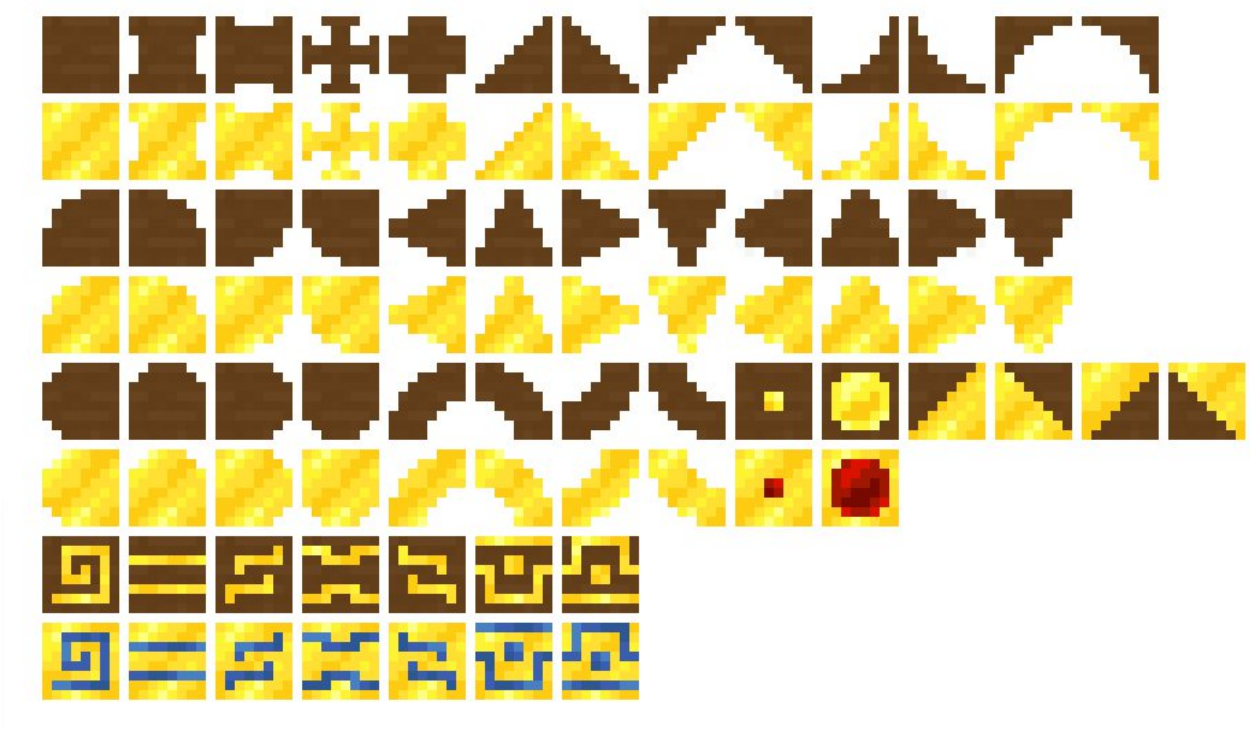


Рисунок 2.6 - Кінцевий набір фрагментів з гри

Маючи намальовані фрагменти, потрібно було повторно перевірити життєздатність ідеї та зайнятись левел-дизайном. В результаті було створено 13 рівнів. Всі вони відображають якусь існуючу фігуру та мають відповідну назву: "Cup", "Bottle", "Planter", "Tazza", "Vase", "Jug", "Pot", "Amphora", "Urn", "Crock", "Large bottle", "Kalyx krater", "Gold Lamp" (рис. 2.7).

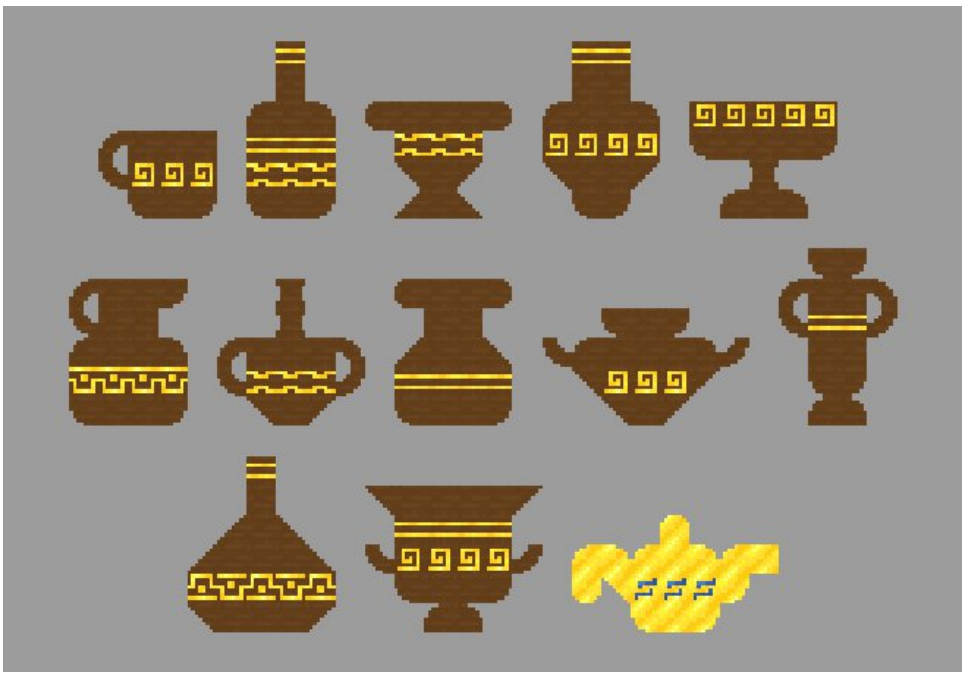


Рисунок 2.7 - Кінцевий набір фігур гри

Наступним кроком стало планування внутрішньої навігації гри (рис 2.8). Було сформовано ряд основних меню та екранів, серед яких: головне меню, меню рівнів, власне рівні, екран паузи, екран програшу, екран проходження рівня, екран налаштувань та мануал.

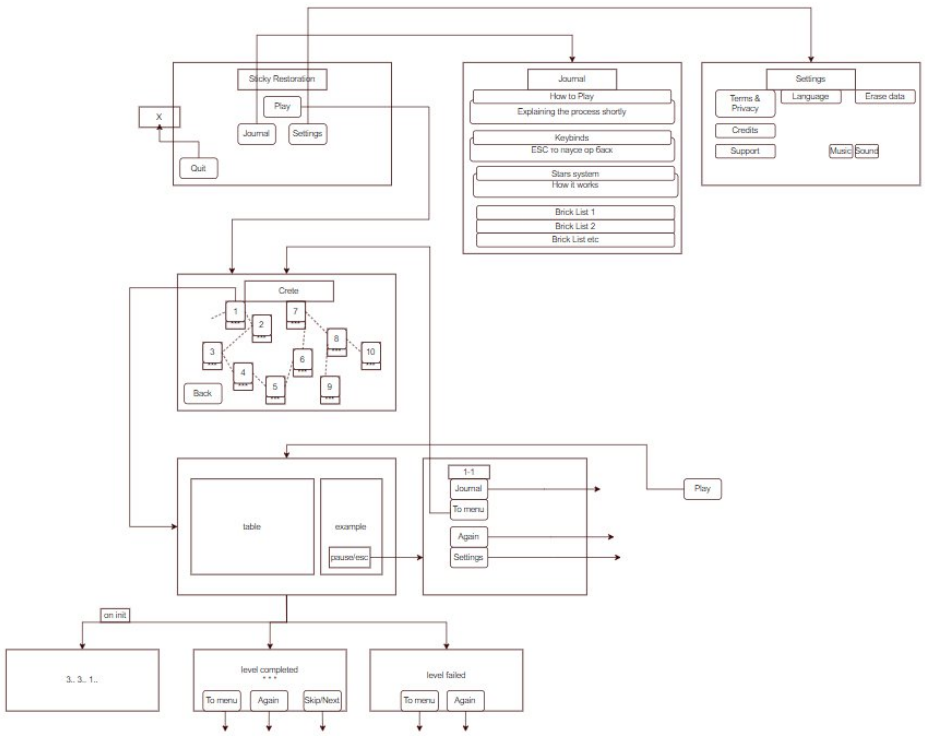


Рисунок 2.8 - Початковий начерк шляхів навігації між екранами у грі

Дизайн рівня гри було виконано згідно до обраного стилю та сеттінгу. Поле гри було намальоване у вигляді стола зі скатертиною, поле прикладу у вигляді меншого стола з сувоєм та пустинне оточення у вигляді піску з камінням. Скатертина спеціально була намальована клітчатою, для досягнення кращої навігації під час гри. Також для гри було спеціально розроблено та намальовано набір гліфів неіснуючої мови, які будуть розміщуватися на сувої, поруч з прикладом. Фрагменти прикладу були намальовані в одній палітрі з гліфами, щоб відобразити, що фігура знаходиться намальованою на сувої. Фрагменти, які слід підібрати, були виокремлені за допомогою піщинок, зображених поверх звичної текстури.

Половина сувою була відведена під назву рівня, відлік часу та новий елемент, який добавився на цьому етапі – зірки, з зображеними на них умовами їх отримання. Для всіх надписів було використано пікселізовану версію шрифту “Mouldy Cheese Regular”.

Також, для кращого занурення у сеттінг, кнопка паузи була зображена як один із каменів на піску (рис. 2.9).

Рисунок 2.9 - Кінцевий вигляд рівня гри

З технічної частини було реалізовано рух гравця, підбирання фрагментів, сповільнення структури гравця на краях ігрового поля і знищення фрагментів при виході за ділянку ігрового поля. Також не підібрані фрагменти зникають з часом, подаючи характерні сигнали перед зникненням. Було добавлено подію програшу та виграшу згідно з попередньо описаним концептом.

При програші, гравцю висвітлюється відповідний екран, який повідомляє, що рівень не пройдено, та містить кнопку переходу до меню рівнів та кнопку “спробувати ще раз” (рис 2.10).

Рисунок 2.10 - Рівень не пройдено

При виграші гравцю висвітлюється відповідний екран, який повідомляє, що рівень пройдено, містить інформацію про те, на скільки добре його пройдено, показує час проходження рівня, кількість фрагментів, що з’являлися протягом гри та кількість підібраних фрагментів. На цьому екрані є кнопка переходу до меню рівнів, кнопка “спробувати ще раз” та кнопка переходу до наступного рівня, що теж веде в меню рівнів (рис 2.11).

Рисунок 2.11 - Рівень пройдено

Натискаючи кнопку паузи в процесі гри, відкриється відповідне меню (рис. 2.12). В меню паузи можна побачити назву рівня, кнопку, що веде у меню рівнів, кнопку, що веде в екран налаштувань, кнопку “почати заново” та стилізовану під блокнот кнопку відкриття мануалу. Також, кнопка паузи змінює облік з кам’яного на синій.

Рисунок 2.12 - Меню паузи відкрите під час ігрового процесу

Натискаючи кнопку налаштувань в меню паузи чи в головному меню, відкриється відповідний екран (рис 2.13), що містить такі налаштування:

- “Delimiters” – включає та виключає сітку, що виокремлює кожен фрагмент фігури
- “Pace Slowing” – дозволяє сповільнювати рух за допомогою натиску та утримування пробілу
- “Erase Data” – видаляє всі дані прогресу користувача, ця опція не доступна, якщо екран відкрито в процесі гри.

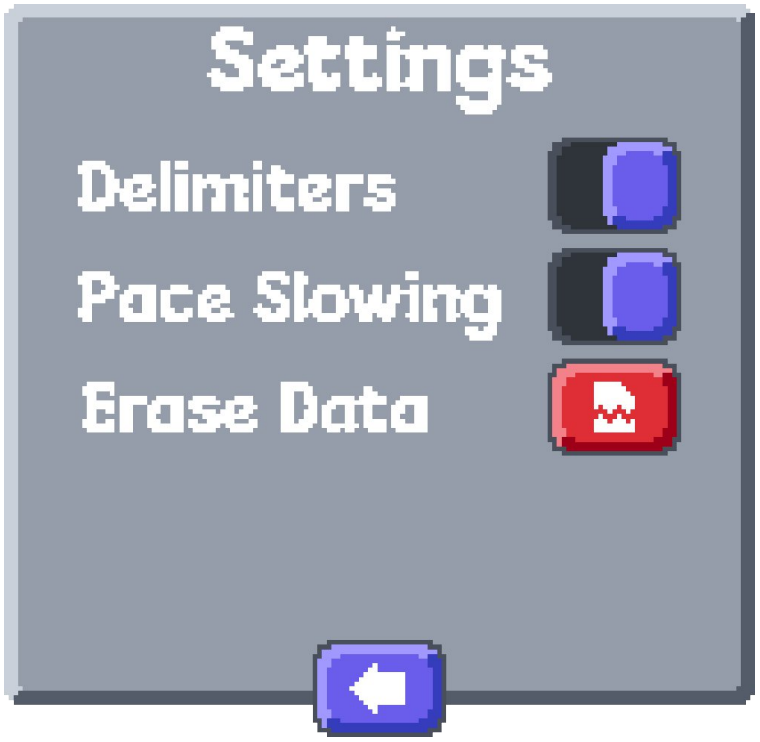


Рисунок 2.13 - Екран налаштувань

Натискаючи кнопку інформації в головному меню чи кнопку мануалу в меню паузи, відкриється відповідний екран, стилізований під відкритий записник (рис 2.14), що містить коротку інформацію про геймплей для початківця.

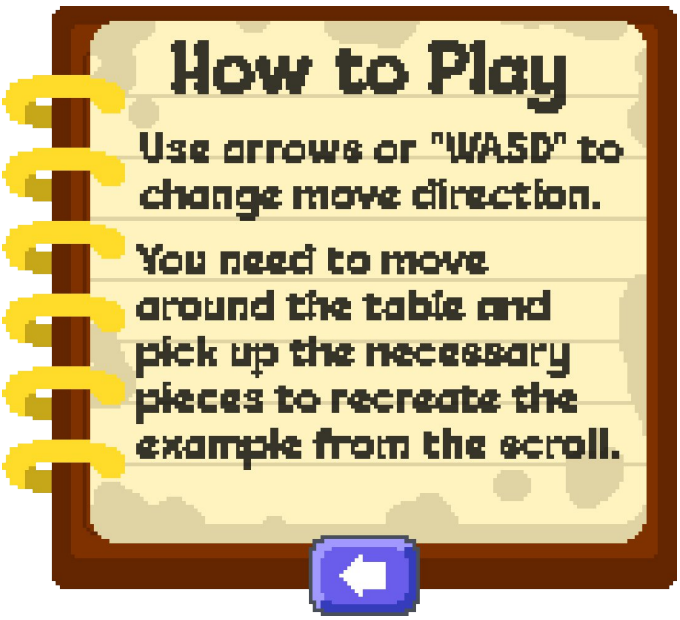


Рисунок 2.14 - Екран мануалу

Наступним кроком було створення меню рівнів, яке згідно задуму мало виглядати як полиця з фігурами пройдених рівнів. Невідкриті рівні позначаються тінню зі знаком запитання, відкритий але не пройдений рівень був зображений у формі великої синьої кнопки з символом “play”, а пройдений ілюструє ту чи іншу фігуру та кількість зірок, на яку рівень було пройдено (рис 2.15). При наведенні на пройдений рівень, він стає активним і з’являється тонка синя кнопка із символом “play”, при натиску, гравця переносить на обраний рівень.

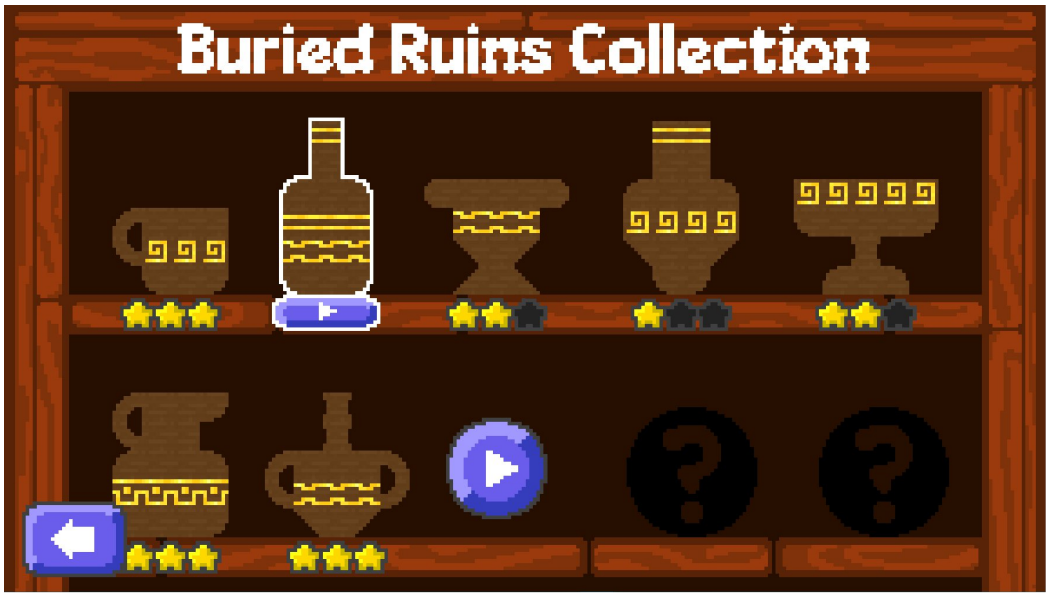


Рисунок 2.15 - Кінцевий вигляд меню рівнів гри.

Далі було добавлене головне меню, на якому було розміщена назва гри, фон у вигляді розкопаної піщаної поверхні та кнопки у вигляді розкопаних фрагментів, призначення кнопок було зображено у формі, візерунка на фрагментах. В меню розміщені кнопки “play”, що веде в меню рівнів, кнопка налаштувань, кнопка інформації та кнопка виходу з гри (рис 2.16)

Рисунок 2.16 - Кінцевий вигляд головного меню гри.

2.4 Обґрунтування вибору інструментальних засобів розробки

У процесі розробки гри було використано низку інструментальних засобів, які були обрані з урахуванням їх функціональності, зручності використання та доступності. Ключовими інструментами, використаними в цьому проєкті, є:

В якості рушія було вибрано Godot 4.2.1. Godot є безкоштовним відкритим інструментом з великою спільнотою розробників та активною підтримкою. Крім того, актуальна версія 4.2.1. має значні покращення в структурі рушія та нові функції, які спрощують розробку ігор. Godot також є зручним у використанні та

має низькі системні вимоги, що робить його привабливим вибором для розробників різного рівня.

Весь код гри був написаний на мові програмування GDScript. Це вбудована мова програмування в Godot, спеціально розроблена для цього рушія. Вона має синтаксис, подібний до Python, що робить її легкою у вивченні та використанні. GDScript тісно інтегрований з Godot, забезпечуючи безперебійну взаємодію між кодом та редактором рушія, що дозволяє ефективно реалізовувати логіку гри.[29]

Для створення графічних елементів та редагування зображень було використано безкоштовний растровий графічний редактор Paint.NET. Редактор має легкий та інтуїтивно-зрозумілий інтерфейс, широкий набір інструментів та підтримку .dll плагінів, що дозволяє розширювати його функціональність до будь-яких масштабів. Paint.NET є легким у використанні та достатньо потужним для задоволення потреб у створенні графіки для гри. [30]

В процесі розробки гри часто потрібно було реалізовувати діаграми, мокапи та схематичні зображення інтерфейсу. Для цього було використано онлайн-інструмент Diagrams.net. Цей безкоштовний ресурс дозволяє легко створювати різноманітні схеми, діаграми та макети, забезпечуючи зручний інтерфейс та широкий набір форм і фігур. [31]

Для отримання звукових ефектів та музичного супроводу для гри було використано ресурс freesound.org. Ця платформа надає величезну бібліотеку аудіо-файлів, з безкоштовними для комерційного використання ліцензіями, що дозволяє легально використовувати їх у проєктах. [32]

Для контролю версій коду під час розробки гри було використано систему контролю версій Git та хмарний сервіс GitHub. Git є розподіленою системою контролю версій, яка дозволяє ефективно відстежувати зміни в коді, створювати гілки розробки, об’єднувати їх та повертатися до попередніх версій у разі необхідності. GitHub в свою чергу забезпечує безпечне хмарне сховище для коду та надає зручний веб-інтерфейс для керування репозиторіями.

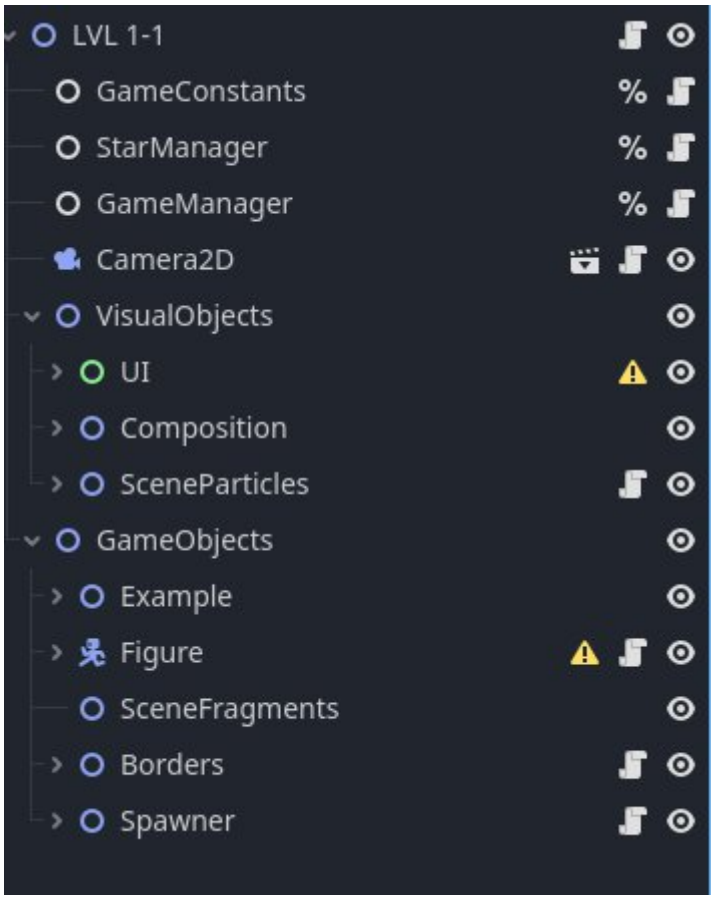
Для організації дизайн-документації, створення попунктових планів та ведення заміток було використано програму Obsidian. Це потужний інструмент

для керування базами знань, який дозволяє структурувати та зв’язувати інформацію за допомогою гіпертексту, забезпечуючи зручний доступ та навігацію в документах. [33]

Кожен із цих інструментів був ретельно підібраний з урахуванням його переваг, функціональності та відповідності потребам розробки гри. Їх комбінація дозволила ефективно реалізувати задум проєкту та забезпечити високу якість кінцевого продукту.

2.5 Особливості програмної реалізації

Основна структура рівня поділена на менеджер-частину, візуальні об’єкти та ігрові об’єкти. (рис. 2.17)



2.17 - Основна структура рівня

Вузол GameConstants відповідає за збереження всіх заданих розробником налаштувань для обраного рівня. Вузол визначає властивості рівня через експортовані змінні, такі як розділ, тему, назву, умови отримання зірок. Також тут задаються налаштування фрагментів – типи, кількість, координати атласу. Встановлюються відступи та розміри ігрового поля, час життя фрагментів та інтервал спауну. (рис. 2.18)

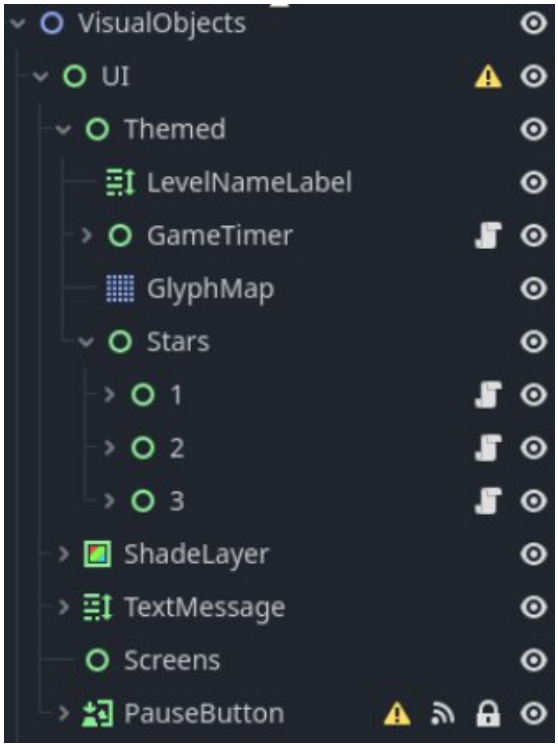
```gdscript
@export_category("Level Properties")
@export var chapter: GlobalManager.Chapters
@export var level_theme: GlobalManager.Themes
@export var level_name: String
@export var figure_name: String
@export_group("Stars")
@export var n1_star_condition: GlobalManager.StarConditions
@export var n2_star_condition: GlobalManager.StarConditions
@export var n3_star_condition: GlobalManager.StarConditions
@export_group("Fragments")
@export var first_tile_atlas_cords: Vector2i
@export var min_fragment_spawn_count: int = 2
@export var max_fragment_spawn_count: int = 6
@export_subgroup("Fragment Set")
@export var allowable_fragments: Array[GlobalManager.FragmentTypes]
@export var fragment_quotes: Array[int]
@export var critical_fragment: GlobalManager.FragmentTypes
@export_group("Game Field")
@export var spawn_padding: Vector2 = Vector2(5, 5)
@export var game_padding: Vector2 = Vector2(3, 3)
@export_group("Time")
@export var fragment_life_time_range: Vector2 = Vector2(14,16)
@export var spawner_wait_time: float = 10
#External
@onready var game_field = $"../VisualObjects/Composition/GameField"
#Main vars
var spawn_area: Array[Vector2]
var game_action_area: Array[Vector2]
var clear_game_area: Array[Vector2]

var slow_border_positions: Dictionary

func _ready():
    spawn_padding = spawn_padding * GlobalManager.TILE_SIZE
    game_padding = game_padding * GlobalManager.TILE_SIZE
    var game_field_start  = game_field.get_rect().position + game_field.position
    var game_field_end  =  game_field.get_rect().end + game_field.position - Vector2(GlobalManager.TILE_SIZE, GlobalManager.TILE_SIZE)
    var game_field_true_end = game_field.get_rect().end + game_field.position
    spawn_area = [game_field_start + spawn_padding, game_field_end - spawn_padding]
    game_action_area = [game_field_start + game_padding, game_field_end - game_padding]

    clear_game_area = [game_field_start, game_field_true_end]
    var middle_x = (game_field_start.x + game_field_true_end.x) / 2
    var middle_y = (game_field_start.y + game_field_true_end.y) / 2
    slow_border_positions = {
        "up": Vector2(middle_x, game_field_start.y),
        "down": Vector2(middle_x, game_field_true_end.y),
        "left": Vector2(game_field_start.x, middle_y),
        "right": Vector2(game_field_true_end.x, middle_y)
    }
```

2.18 - Скрипт вузла GameConstants

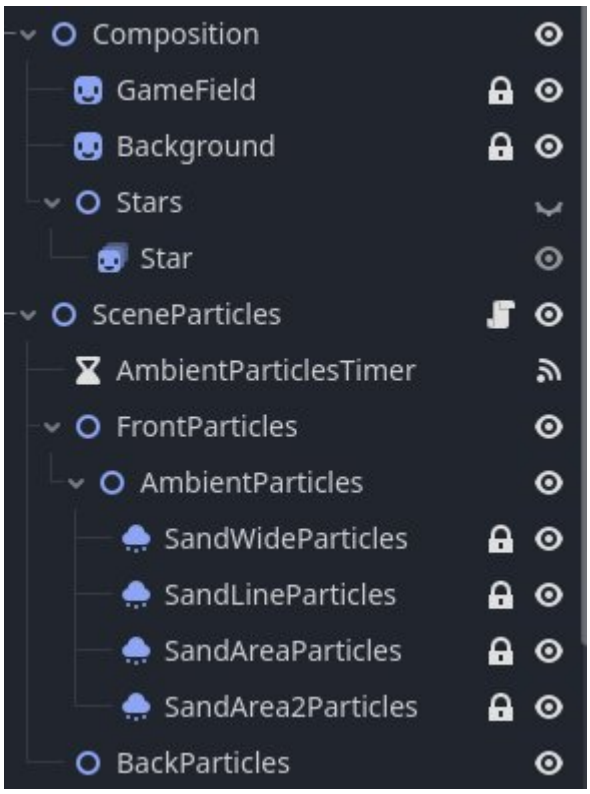
У вузлі GameManager реалізована машина станів рівня. Цей вузол є контролером стану гри. Він керує переходами між різними станами гри (початок, гра, пауза, перемога, програш) через спеціальні функції-обробники. Також відповідає за відображення відповідних екранів/повідомлень, призупинення/відновлення гри та перевірку умов перемоги/програшу порівнянням сформованої фігури з прикладом. Ініціалізує візуальні елементи інтерфейсу відповідно до налаштувань рівня.

Вузол StarManager відстежує стан умов для отримання зірок на рівні та відповідає за візуалізацію зірок. Зберігає масив станів умов зірок, загальну кількість зірок. Ініціалізує візуальні елементи зірок та часових відміток таймера згідно заданих умов.

Вузол VisualObjects містить в собі UI-частину (рис 2.19), частинки сцени та композицію (рис 2.20) В UI-частині Свої скрипти мають зірки та таймер, які здатні оновлювати свій вигляд, взаємодіючи з StarManager. В частині композиції знаходиться основне статичне наповнення рівня, а частина з частинками сцени відведена під всі частинки, що з’являтимуться на рівні.



2.19 - UI-частина вузла "VisualObjects"



2.20 - Частина частинок сцени та композиції "VisualObjects"

Вузол "GameObjects" містить ігрові об’єкти та їхні компоненти (2.21), зокрема:

- Example – вузол з прикладом фігури, що складається з ExampleMap (тайлсет прикладу) та DelimiterMap (карта обмежень).
- Figure – вузол фігури гравця, яка є основним елементом гри та складається з FigureMap (тайлмапу фігури), ShadeMap (тайлмапу тіней), DelimiterMap (тайлмапу розділювачів).
- SceneFragments – вузол для заспаунених фрагментів.
- Borders – вузол з кордонами ігрового поля.
- SlowBorderH/V – горизонтальні та вертикальні кордони для сповільнення.
- CutBorderH/V – кордони для обрізання фігури при входженні в їх область.
- KillBorderH/V – резервні кордони для повного знищення фігури.
- Spawner – вузол-спаунер фрагментів з таймером SpawnTimer.

- HandStructure – структура руки, яку викликає спаунер для спауну фрагментів.

2.21 - Вузол "GameObjects"

Фігура гравця реалізована як тайлмап (TileMap), що є ефективним рішенням у випадку особливої логіки цієї гри. Використання тайлмапу для фігури гравця дозволяє спростити обробку зіткнень, візуалізацію та маніпуляції з окремими плитками фігури. Це зменшує навантаження на обчислювальні ресурси, оскільки тайлмапи розроблені спеціально для ефективного відображення та керування великою кількістю однорідних елементів на основі плиток. Замість того, щоб створювати окремий вузол для кожної плитки фігури,

тайлмап дозволяє працювати з усією фігурою як з одним об’єктом, легко змінюючи окремі плитки за допомогою простих функцій та операцій над комірками. Це спрощує код, підвищує продуктивність та дозволяє ефективно керувати великими фігурами, складеними з багатьох плиток. (рис. 2.22)

```
func _on_collide_data_send(fragment, body):
    game_manager.collided_fragments_count += 1
    if body == figure_map:
        match fragment.type:
            GlobalManager.FragmentTypes.DEFAULT:
                default_fragment_collide(fragment, body)
            GlobalManager.FragmentTypes.BOMB:
                bomb_fragment_collide(fragment, body)
            #GlobalManager.FragmentTypes.THORN:
            _:
                pass
        GlobalManager.emit_signal("end_game_check")

func bomb_fragment_collide(fragment, body):
    var local_coords : Vector2 = body.to_local(fragment.action_marker.global_position)
    var tile_coords = body.local_to_map(local_coords)
    delete_cell(tile_coords) ##NO!

func default_fragment_collide(fragment, body):
    var local_coords : Vector2 = body.to_local(fragment.vision_box.global_position)
    var tile_coords = body.local_to_map(local_coords)
    splash_emit(local_coords)
    #var surrounding_cells = body.get_surrounding_cells(tile_coords)
    figure_map.set_cell(0, tile_coords, 0, fragment.texture_atlas_cords)
    shade_map.set_cell(0, tile_coords, 0, Vector2i(fragment.texture_atlas_cords.x, 0))
    delimiter_map.set_cell(0, tile_coords, 0, Vector2i(fragment.texture_atlas_cords.x, randi_range(0, 2)))
```

Рисунок 2.22 - Код реагування на дотик гравця до фрагментів різного типу

2.6 Організація тестування та налагодження

У процесі розробки відеогри ретельне тестування проводилося на різних етапах створення гри, згідно з циклом обраної ітераційної моделі. Тестування відбувалося як самостійно розробником, так і за участі інших людей, які виступали в ролі бета-тестерів.

На початкових стадіях розробки основний акцент було зроблено на тестуванні базових ігрових механік шляхом відтворення як очікуваних, так і максимально неочікуваних ігрових сценаріїв. Ця частина тестування допомогла виявити та усунути основні критичні помилки.

Окрім функціональної складової, велике значення мало тестування користувацького інтерфейсу та зручності гри. Для цього було залучено групу з 3 бета-тестерів різного віку та рівня навичок у відеоіграх. На основі зібраних від бета-тестерів відгуків було проведено низку виправлень та вдосконалень інтерфейсу, графіки, керування, ігрового балансу та інших користувацько-орієнтованих аспектів гри.

Загалом, процес тестування та налагодження ігрового програмного забезпечення був комплексним і здійснювався на різних рівнях – від перевірки базової функціональності до тестування користувацького досвіду. Такий підхід дозволив усунути більшість потенційних помилок та недоліків.

2.7 Рекомендації по використанню та впровадженню програмного засобу

Розроблена відеогра має потенційно широкий спектр застосування та може бути корисною у різних сферах людської діяльності. Основною цільовою аудиторією є діти та підлітки.

Перш за все, гра рекомендована для розваг та проведення вільного часу, проте також може використовуватись для навчальних та розвиваючих цілей.

Наявність рівнів різної складності та нетиповий ігровий процес роблять її цікавою для груп різних вікових категорій. Дослідницька атмосфера та яскрава графіка сприяють позитивному емоційному стану під час гри.

Також гру можна використовувати в якості навчального компоненту для розвитку логічного та просторового мислення у дітей. Необхідність аналізувати форми, планувати послідовність кроків та передбачати їх наслідки сприяє тренуванню важливих розумових навичок.

ВИСНОВКИ

У даній роботі було досліджено поняття відеоігор, їх історію, особливості жанрів та ігрових механік. Проаналізовано існуючі рішення та приклади гібридизації жанрів у сучасних відеоіграх.

На основі проведеного аналізу було розроблено концепцію комп’ютерної гри "Piece by Piece" з геймплейними механіками, що поєднують елементи пазлу, аркади, екшну та стратегії. Основна ідея гри полягає у формуванні гравцем певної структури на ігровому полі відповідно до наданого зразка шляхом приєднання різних фрагментів.

Реалізація розробленої концепції була здійснена в ігровому рушії Godot 4. Було створено повноцінний програмний продукт, що містить всі елементи, необхідні для справного функціонування.

Розроблений програмний засіб демонструє успішне поєднання різних ігрових жанрів та може слугувати основою для подальшого розвитку та масштабування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Contributors to Wikimedia projects. Video game - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Video_game (date of access: 05.06.2024).

2. Contributors to Wikimedia projects. Cathode-ray tube amusement device - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Cathode-ray_tube_amusement_device (date of access: 05.06.2024).

3. Contributors to Wikimedia projects. Tennis for Two - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Tennis_for_Two (date of access: 05.06.2024).

4. Contributors to Wikimedia projects. Spacewar! - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Spacewar! (date of access: 09.06.2024).

5. Учасники проектів Вікімедіа. Magnavox Odyssey – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Magnavox_Odyssey (дата звернення: 09.06.2024).

6. Contributors to Wikimedia projects. Space Invaders - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Space_Invaders (date of access: 05.06.2024).

7. Contributors to Wikimedia projects. id Software - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Id_Software (date of access: 05.06.2024).

8. Adams E. Fundamentals of game design. 3rd ed. San Francisco : New Riders, 2013.

9. Contributors to Wikimedia projects. Action game - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Action_game (date of access: 05.06.2024).

10. Contributors to Wikimedia projects. Adventure game - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Adventure_game (date of access: 05.06.2024).

11. Учасники проєктів Вікімедіа. Головоломка (жанр відеоігор) – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Головоломка_(жанр_відеоігор) (дата звернення: 05.06.2024).

12. Contributors to Wikimedia projects. Role-playing game - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Role-playing_game (date of access: 05.06.2024).

13. Contributors to Wikimedia projects. Simulation video game - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Simulation_video_game (date of access: 05.06.2024).

14. Contributors to Wikimedia projects. Strategy video game - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Strategy_video_game (date of access: 05.06.2024).

15. Contributors to Wikimedia projects. List of video game genres - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/List_of_video_game_genres (date of access: 05.06.2024).

16. Contributors to Wikimedia projects. Game mechanics - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Game_mechanics (date of access: 05.06.2024).

17. Учасники проєктів Вікімедіа. Ігрова механіка – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Ігрова_механіка (дата звернення: 05.06.2024).

18. WestHabit8792. Mixing Genres, Good or Bad?. Reddit. URL: https://www.reddit.com/r/gamedesign/comments/108e7tv/mixing_genres_good_or_bad/ (date of access: 05.06.2024).

19. Broaden Your Game’s Audience With Hybrid Genres - GameRefinery. GameRefinery. URL: https://www.gamerefinery.com/broaden-your-games-audience-with-hybrid-genres/ (date of access: 05.06.2024).

20. Учасники проєктів Вікімедіа. Ігровий рушій – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Ігровий_рушій (дата звернення: 05.06.2024).

21. Contributors to Wikimedia projects. Game engine - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Game_engine (date of access: 05.06.2024).

22. Contributors to Wikimedia projects. Unity (game engine) - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Unity_(game_engine) (date of access: 05.06.2024).

23. Contributors to Wikimedia projects. Unreal Engine - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Unreal_Engine (date of access: 05.06.2024).

24. Contributors to Wikimedia projects. Godot (game engine) - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Godot_(game_engine) (date of access: 05.06.2024).

25. Features - Godot Engine. Godot Engine. URL: https://godotengine.org/features/ (date of access: 05.06.2024).

26. A Most Extraordinary Gnome. Godot Engine. URL: https://godotengine.org/showcase/a-most-extraordinary-gnome/ (date of access: 05.06.2024).

27. Brotato. Godot Engine. URL: https://godotengine.org/showcase/brotato/ (date of access: 05.06.2024).

28. Возняк, М. А. Розробка інноваційних ігрових механік шляхом мікшування різних жанрових підходів. Матеріали XVIII Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених "Молода наука Волині: пріоритети та перспективи досліджень" (14–15 трав. 2024 р.). Луцьк : Волинський національний університет імені Лесі Українки, 2024. 1 електрон. опт. диск (CD-ROM). С. 330–332. ISBN 978-966-940-565-4.

29. GDScript. GDScript. URL: https://gdscript.com/ (date of access: 05.06.2024).

30. Ласкаво просимо на сайт Paint.NET. Paint.NET. URL: https://paintnet.org.ua/ (дата звернення: 05.06.2024).

31. Contributors to Wikimedia projects. diagrams.net - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Diagrams.net (date of access: 05.06.2024).

32. Freesound. Freesound. URL: https://freesound.org/ (date of access: 05.06.2024).

33. Obsidian - Sharpen your thinking. Obsidian - Sharpen your thinking. URL: https://obsidian.md/ (date of access: 05.06.2024).

ДОДАТКИ

ДОДАТОК А
ТЕХНІЧНЕ ЗАВДАННЯ

1.Вступ

"Piece by Piece" – комп’ютерна гра жанру пазл-аркада з елементами екшену та стратегії, розроблена в ігровому рушії Godot 4.

2.Підстави для розробки

Розроблена програма була створена в рамках кваліфікаційної роботи на здобуття освітнього ступеня «бакалавр».

3.Призначення розробки

Відеогра "Piece by Piece" призначена для розваг, навчання та розвитку логічного і просторового мислення у широкої цільової аудиторії різних вікових категорій, в основному дітей та підлітків.

4. Вимоги до програми чи програмного продукту

4.1 Вимоги до функціональних характеристик

Гра повинна забезпечувати:

- Реалізацію повного ігрового процесу з рухом гравця, взаємодією з різними типами фрагментів, обмеженнями ігрового поля та умовами перемоги/поразки.
- Наявність головного меню, меню рівнів з можливістю вибору та запуску окремих рівнів.
- Екран налаштувань для налаштування візуальних та ігрових параметрів.
- Екран інструкцій з детальним поясненням геймплею.
- Екрани виграшу та програшу з відповідною інформацією та варіантами подальших дій.
- Екран паузи з можливістю керування ігровим процесом.
- Інтуїтивно зрозумілий інтерфейс із зручною навігацією.
- Коректну обробку помилок та виняткових ситуацій.

4.2 Вимоги до надійності

Програма повинна працювати стабільно та без критичних помилок на цільових платформах.

4.3 Умови експлуатації

Для використання програми необхідно та достатньо мати лише її .exe-файл.

4.4 Вимоги до складу і параметрів технічних засобів

Програма функціонує на персональних комп’ютерах операційної системи Windows 10.

4.5 Вимоги до інформаційної і програмної сумісності

За умов дотримання вимог до складу і параметрів технічних засобів, вимог до інформаційної і програмної сумісності не виникає.

4.6 Вимоги до маркування і упаковки

Немає вимог до маркування і упаковки.

4.7 Вимоги до транспортування і збереження

Вимоги до транспортування та зберігання відсутні.

5 Вимоги до програмної документації

Програмна документація повинна включати технічне завдання та керівництво користувача.

6 Техніко-економічні показники

Програмний продукт не передбачає жодних витрат з боку користувача.

7 Стадії і етапи розробки

Етапи розробки відображені в наступних пунктах:

1. Дослідження обраної теми та загальних відомостей з неї, основ створення відеоігор, особливостей жанрів та ігрових механік.

2. Розробка концепції гри, визначення ігрових механік та елементів різних жанрів, що будуть поєднані. Створення дизайнерських та алгоритмічних рішень.

3. Розробка програмного продукту, втілення дизайну та ігрових механік в ігровому рушії Godot 4. Паралельне тестування на кожній ітерації розробки.

4. Фінальне комплексне тестування гри та внесення необхідних коригувань і доопрацювань.

8 Порядок контролю і приймання

Програма повинна бути розроблена в часових межах, виділених під реалізацію кваліфікаційної роботи на здобуття освітнього ступеня «бакалавр» та прийнята в день її здачі разом з усіма супутніми матеріалами.

ДОДАТОК Б
КЕРІВНИЦТВО КОРИСТУВАЧУ

Загальні відомості

Найменування гри: "Piece by Piece".

Функціональне призначення

Програмний засіб призначений для виконання наступних задач:

- Надання користувачу очікуваного та справного повного ігрового процесу.
- Реалізацію ігрового процесу з керуванням рухом гравця, взаємодією з різними типами фрагментів, формуванням заданої структури та умовами перемоги/поразки.
- Наявність головного меню з доступом до налаштувань, інструкцій та вибору рівнів.
- Забезпечення зручного інтерфейсу користувача з можливістю призупинення/відновлення гри.
- Відображення необхідної ігрової інформації та реагування на дії користувача.

Умови застосування програми

Для запуску гри необхідний комп’ютер або ноутбук з операційною системою Windows 10 і вище, процесор з частотою 1.4 ГГц, або більш потужний, ОЗП об’ємом 500 МБ, 500 МБ доступного простору на жорсткому диску.

Повідомлення оператору та опис роботи програми

При запуску гри, перед користувачем відкривається головне меню (рис. 3.1)

Рисунок 3.1 - Схема головного меню

1 – кнопка переходу до меню рівнів

2 – кнопка для показу меню інформації

3 – кнопка для показу меню налаштувань

4 – кнопка виходу з гри

Меню рівнів містить доступ до всіх рівнів у тому чи іншому вигляді (рис. 3.2).

Рисунок 3.2 - Схема меню рівнів

1 – активний розблокований пройдений рівень (наведено курсором)

2 – розблокований пройдений рівень

3 – не розблокований рівень

4 – розблокований не пройдений рівень

5 – кнопка повернення в головне меню

Ігровий рівень виглядає наступним чином (рис. 3.3).

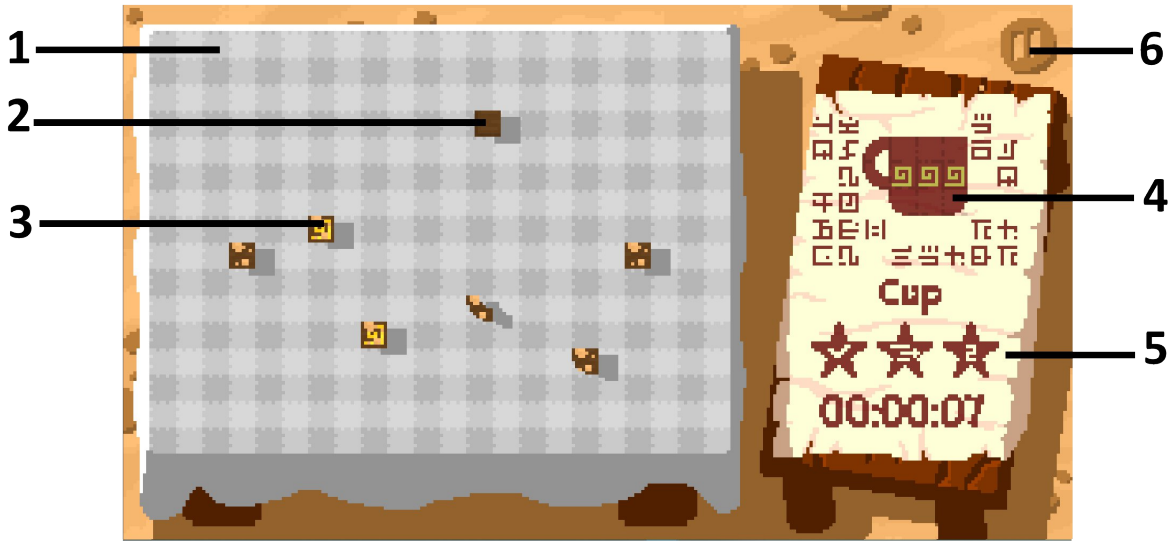


Рисунок 3.3 - Схема рівня гри

1 – поле гри

2 – гравець на початку гри

3 – один із фрагментів, що можна підібрати

4 – приклад, який потрібно відтворити

5 – інформація про рівень та його проходження

6 – кнопка паузи щ овикликає відповідне меню

Меню паузи має наступний вигляд (рис. 3.4).

Рисунок 3.4 - Меню паузи

1 – кнопка для показу меню інформації

2 – кнопка переходу до меню рівнів

3 – кнопка перепроходження рівня заново із втратою прогресу

4 – кнопка переходу в екран налаштувань

В екрані налаштувань доступні наступні налаштування гри (рис. 3.5).

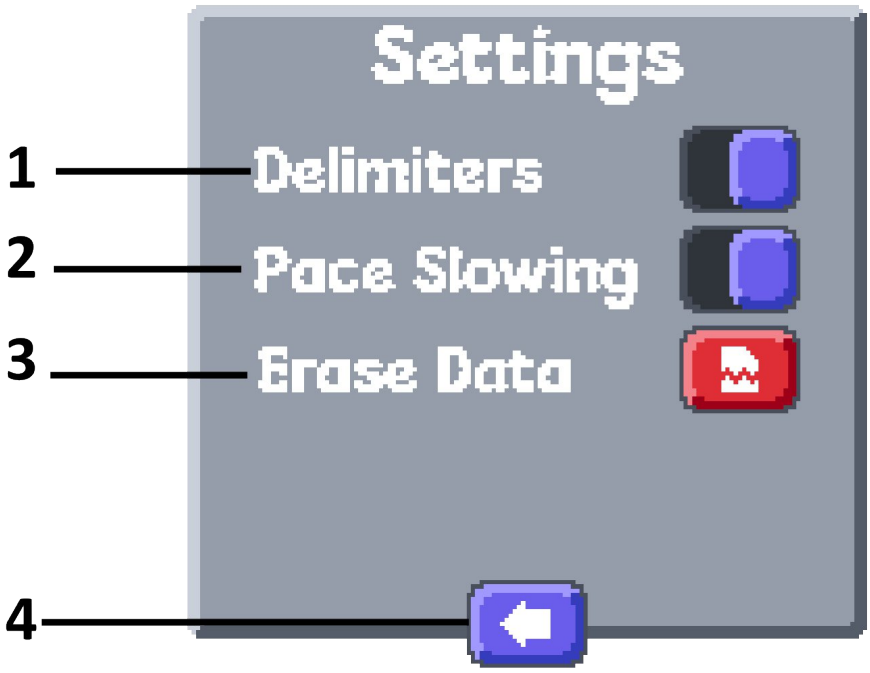


Рисунок 3.5 - Екран налаштувань

1 – увімкнення роздільників (внутрішньоігровий елемент)

2 – увімкнення можливості сповільнення темпу (внутрішньоігровий елемент)

3 – кнопка очищення всього ігрового прогресу

4 – кнопка виходу з налаштувань

АНОТАЦІЯ

Возняк М. А. – **Дослідження можливостей рушія Godot для розробки гри комбінованих жанрів** – Рукопис.

Кваліфікаційна робота за спеціальністю 122 Комп’ютерні науки. – Волинський національний університет імені Лесі Українки, Луцьк. – 2024р.

Робота присвячена розробці комп'ютерної гри "Piece by Piece", що поєднує елементи різних ігрових жанрів: пазл, аркада, екшн та стратегія. Особлива увага приділяється створенню унікальних ігрових механік та їх реалізації за допомогою ігрового рушія Godot 4. Мета роботи: розробка комп'ютерної гри з інноваційними геймплейними механіками шляхом поєднання елементів різних жанрів. Гра пропонує нетривіальний підхід до ігрового процесу, де основним завданням є реконструкція заданих структур з динамічно генерованих фрагментів на обмеженому ігровому полі. Робота охоплює всі етапи створення гри: від концептуалізації та дизайну до програмної реалізації та тестування.

Ключові слова: розробка відеоігор, гібридизація жанрів, інноваційні ігрові механіки, Godot 4, піксельна графіка, Баухаус, пазл, аркада, екшн, стратегія, геймдизайн, левел-дизайн.