Міністерство освіти і науки України
Кам’янець-Подільський національний університет імені Івана Огієнка
Фізико-математичний факультет
Кафедра комп’ютерних наук

Дипломна робота
магістра
з теми: «**СТВОРЕННЯ ІГРОВИХ ДОДАТКІВ З ВИКОРИСТАННЯМ 3D-ГРАФІКИ ДЛЯ ОС ANDROID**»

Виконав: студент 2 курсу,
групи KN1-M21,
спеціальності 122 Комп’ютерні науки
Груша Микола Миколайович

Керівник: **Пилипюк Т.М.**,
кандидат фізико-математичних наук,
доцент, доцент кафедри комп’ютерних наук

Рецензент: **Громик А.П.**,
кандидат технічних наук, доцент, завідувач кафедри математики, інформатики та академічного письма закладу вищої освіти «Подільський державний університет»

Кам’янець-Подільський – 2022

ЗМІСТ

ВСТУП

Мобільні відеоігри сьогодні – це один з перспективних напрямків на ринку медіа-розваг. Ігри стали займати велику кількість часу багатьох користувачів персональних комп'ютерів і мобільних пристроїв. Через стрімкий розвиток цієї індустрії розробка мобільних відеоігор все більше сприймається розробниками і геймерами як створення мистецтва.

Ринок відеоігор величезний, в якому знаходять місце як малі студії так і великі, а також видавці, які допомагають створювати високобюджетні продукти, що з кожним роком продаються все більшими копіями.

Відеоігри мають такий рівень популярності, що випускаються масово. Щоб збільшити масовість продукту використовують фокус-групи багато видавців опираються на них задля збільшення продажів. Це призводить до великої кількості проектів, які схожі один на одного. Відкриття цифрових магазинів допомогло розповсюдженню відеоігор, дозволивши незалежним розробникам мати можливість вести розробку без ризику, пов'язаного з витратами на створення фізичних копій гри. Це дозволило створювати більш цікаві і ризиковані проекти, на які б великі студії не наважилися.

Інді-розробка відеоігор це не є щось новим в індустрії, але за останні роки вона отримала великий поштовх. Такі розробники все частіше стають популярними, в результаті чого інді-проети мають великий вплив в індустрії відеоігор.

Актуальність теми роботи викликана збільшенням популярністю відеоігор на мобільних платформах. Відеоігри вже давно стали невід'ємною частиною нашого життя. А разом зі зростанням популярності ринку мобільних гаджетів, так само швидко почала зростати популярність ринку цифрових розваг. Тому сьогодні, відеоігрова індустрія є найбільшою частиною світового ринку цифрового розважального контенту, яка кожен рік залучає величезну аудиторію й отримує чисельні доходи.

Предмет дослідження – технології розробки комп'ютерних ігор.

Об'єктом дослідження є розробка мобільних додатків з використанням 3D-графіки для ОС Android.

Мета роботи – створення прототипів мобільних додатків з використанням сучасних засобів створення 3D-графіки в порівнянні можливостей ігрових рушіїв Godot Engine і Unity 5.

Для досягнення поставленої мети потрібно вирішити наступні завдання:

- здійснити аналіз предметної області;
- провести аналіз ігрових рушіїв для розробки 3D ігрових додатків;
- дослідити та порівняти можливості рушіїв Unity та Godot;
- здійснити практичну реалізацію ігрових прототипів за допомогою обраних ігрових рушіїв, протестувати та порівняти результати.

Методи дослідження – теоретичні та емпіричні (аналіз та узагальнення досвіду використання ігрової механіки); – системно-функціональні та моделювання (створення структурно-функціональної моделі процесу розробки мобільного ігрового додатку, створення моделі ігрового додатку).

Практичне значення. Відеоігри в більшості випадків використовують для розваг, але також вони допомагають в розвитку моторики, концентрації уваги, стратегічного та креативного мислення у гравця. Ще одна з переваг використання серйозних відеоігор в засвоєнні нових когнітивних навичок, тому, що розвивається логічне мислення та просторове відчуття, навички співпрацювати, що є подальшою підготовкою до розв’язання різних проблемних ситуацій.

Апробація результатів. За темою магістерського дослідження подано статтю «Використання функціоналу програмного пакету Blender для створення та редагування 3D-графіки» до Вісника Кам'янець-Подільського національного університету імені Івана Огієнка. Фізико- математичні науки.

Структура роботи. Дипломна робота складається зі вступу, трьох розділів, висновків та списку використаних джерел.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Поняття та класифікація комп'ютерних ігор

Відеогра – гра, яка являє собою комп’ютерну програму та використовує мультимедійні можливості комп’ютера. Пристрої, які використовують для відеоігор – ігрові платформи, до яких входять: персональний комп'ютер (ПК), консолі, смартфони та ін. Пристрої, які використовують для керування відеоіграми – ігрові контролери, до них входять: геймпади, мишки, клавіатури, сенсорні екрани та ін.

Число жанрів відеоігор має велику кількість, і часто вони класифікуються за характеристиками або завданням. Тому категорії ігор, або жанри, можуть поділятися на піджанри, а одна гра цілком може належати до кількох жанрів.

Розглянемо ці жанри та надамо їх коротку характеристику.

Аркада – жанр відеоігор, ігровий процес (геймплей), яких заснований на швидкої реакції гравця, мінімальному ступені свободи керованих персонажів і простому управлінні. Перші представники Battle City, Pac-Men, Snake.

Ігровий процес в цих іграх простий і короткий – одна ігрова сесія займає всього кілька хвилин, однак гравець не сумуватиме ні на секунду. Завдання можуть бути різні, проте аркадні ігри об'єднує простота в керуванні і цілі – зібрати певну або найбільшу кількість очків, знищити всіх ворогів на рівні, або дістатися до виходу.

Технології, які використовуються:

– проста графіка;
– цікавий геймплей.

Шутери від першої особи (FPS) – жанр, який вводиться такими іграми, як Quake, Unreal Tournament, Half-Life, Counter-Strike та Call of Duty. Ці ігри історично включали порівняно повільний пішохідний перехід потенційно великого, але насамперед коридорного світу. Однак сучасні шутери від

першої особи можуть проводитись у найрізноманітніших віртуальних середовищах, включаючи великі відкриті та конфініковані приміщення. Сучасна механіка руху FPS може включати в себе пішохідний рух, можливість перевезення гравця через наземний транспорт або повітряний транспорт, і навіть через наводний транспорт.

Шутери від першої особи, як правило, фокусуються на таких технологіях, як:

- ефективна візуалізація великих 3D-віртуальних світів;
- точна механіка управління / прицілювання камери;
- анімації з високим рівнем деталізації віртуальної зброї та рук гравця;
- широкий асортимент потужного ручного озброєння;
- модель поведінки камери, що згладжує різкість зіткнення гравця з різноманітними об'єктами;
- великий набір анімації та пророблений інтелект для не ігрових персонажів (вороги та союзники гравця);
- можливість гри через Інтернет (як правило, сервери підтримують до 64 гравців одночасно) та класичний для цього жанру режим гри «Deathmatch».

Платформери та інші ігри від третьої особи (3RD PERSON).

«Платформер» – це термін, що застосовується до персонажних екшн-ігор від третьої особи, де стрибки з платформи на платформу є основним механізмом ігрового процесу. Типові ігри з епохи 2D включають Space Panic, Donkey Kong, Pitfall та Super Mario Brothers. Епоха 3D включає такі платформи, як Super Mario 64, Crash Bandicoot, Rayman 2, The Hedgehog, серії Jak and Daxter, серії Ratchet & Clank та останнім часом Super Mario Galaxy.

Ігри на основі персонажів від третьої особи мають багато спільного з іграми від першої особи, як шутери, але значно більше уваги приділяється здібностям персонажів та руху персонажа в грі. Крім того, для персонажа потрібні високоякісні анімації для всього тіла, на відміну від дещо менш вибагливих вимог до анімації "floating arm" у типовій грі FPS. Важливо

відзначити, що майже всі шутери від першої особи мають онлайн-компонент для гри з іншими гравцями, тому аватар гравця з повним тілом повинен бути наданий на додаток до анімації рук від першої особи.

У платформерах головний герой часто подібний до мультфільмів і не особливо реалістичний або не з високою роздільною здатністю. Однак шутери від третьої особи часто мають дуже реалістичні гуманоїдні моделі гравця. В обох випадках персонаж гравця, як правило, має дуже багатий набір дій та анімації. Деякі з технологій, спеціально орієнтовані на ігри в цьому жанрі, включають:

- платформи, що рухаються, драбини, мотузки, та інші цікаві режими руху;
- головоломки з різними об'єктами у світі;
- камера в режимі від третьої особи залишається зосередженою на персонажі гравця та обертанням якої, як правило, керує гравець за допомогою палички джойстика (на консолі) або миші (на ПК);
- складна система уникнення зіткнення камери з об'єктами у світі, для забезпечення того, щоб точка огляду ніколи не «пробивалась» через геометрію фону або динамічні об'єкти переднього плану.

Файтинги або ігри про боротьбу, як правило, розраховані для двох гравців, в яких персонажі гравців б'ють один одного на рингу. Жанр типізований такими іграми, як Mortal Kombat та Tekken. Традиційно ігри в жанрі файтингу зосередилися на таких технологіях як:

- великий набір анімацій ударів для персонажів;
- точне виявлення ударів;
- система введення користувача, здатна виявляти складні комбінації кнопок і джойстиків;
- натовп або відносно статичний цікавий фон.

Оскільки тривимірний світ у цих іграх малий і камера постійно зосереджена на двох гравцях, історично в цих іграх мало або взагалі не потрібно світового підрозділу або відключення оклюзії. Так само не використовуються передові тривимірні моделі розповсюдження звуку.

Гонки – жанр, який охоплює всі ігри, основним завданням яких є керування автомобілем чи іншим транспортним засобом на якійсь трасі. У жанрі є багато підкатегорій. Готові ігри, орієнтовані на імітацію («симуляцію»), мають на меті забезпечити максимально реалістичний досвід водіння (наприклад, Gran Turismo).

Аркадні гонки віддають перевагу забаві над реалістичністю (наприклад, San Francisco Rush, Cruisin 'USA, Hydro Thunder). Порівняно новий піджанр досліджує субкультуру вуличних перегонів із викрадених транспортних засобів (наприклад, Need for Speed, Juiced).

Гоночна гра часто дуже лінійна, подібно до старих ігор FPS. Однак швидкість подорожі, як правило, набагато швидша, ніж у FPS. Тому більше уваги приділяється дуже довгим коридорам доріжок, або петель, які іноді мають різні альтернативні маршрути та таємні скорочення. Гоночні ігри зазвичай зосереджують усі свої графічні деталі на транспортних засобах, дорозі та найближчому оточенні. Проте картингові гонки також приділяють значну частину візуалізації на анімацію для персонажів, що керують транспортними засобами. Деякі технологічні властивості типової гоночної гри включають такі методи:

- різні види рендерінгу використовуються під час надання віддалених фонових елементів, наприклад, використання двовимірних обьєктів для дерев, пагорбів та гір.
- дорога часто розбивається на відносно прості двовимірні регіони, які називаються "секторами"; ці структури даних використовуються для оптимізації візуалізації та визначення видимості, для побудови маршуту для автомобілів, що не контролюються людьми, та для вирішення багатьох інших технічних проблем.
- камера, як правило, стоїть позаду транспортного засобу з точки зору третьої особи, або іноді знаходиться в середині кабіни.

Стратегія в реальному часі (RTS) – жанр сучасної стратегії в реальному часі (RTS) був визначений грою Dune II: Будівництво династії

(1992). Інші ігри цього жанру включають Warcraft, Command & Conquer, Age of Empires та Starcraft. У цьому жанрі гравець стратегічно розгортає бойові підрозділи у своєму арсеналі через велике ігрове поле, намагаючись перемогти свого опонента. Ігровий світ зазвичай відображається під косим кутом огляду зверху вниз. Гравцю RTS-гри, зазвичай заважають змінювати кут огляду, щоб бачити на великій відстані. Це обмеження дозволяє розробникам використовувати різні оптимізації в механізмі візуалізації гри RTS. Старіші ігри в жанрі використовували структуру світової побудови на основі сітки, а орфографічна проекція була використана для того, щоб значно спростити рендерінг. Сучасні ігри RTS іноді використовують перспективу та справжній тривимірний світ, але вони все ще не можуть не використовувати систему компонування сітки, щоб забезпечити, коректне розміщення бойових одиниць та фонові елементи, такі як будівлі, щоб вони правильно відповідали одна одній. Деякі поширені практики в іграх RTS включають такі методи:

- кожен об'єкт має невелику кількість полігонів, так що гра може підтримувати велику кількість їх на екрані одночасно;
- рельєф місцевості, як правило, має різні рівні, на яких гравці мають більш або менші переваги перед опонентом;
- гравцеві часто дозволяється будувати нові споруди на місцевості на додаток до розгортання своїх сил;
- взаємодія користувачів, як правило, здійснюється за допомогою одного клацання та вибору одиниць в основній області, на додаток є меню або панелі інструментів, що містять команди, обладнання, типи підрозділів, типи будівель тощо.

Ігри з масовим багатокористувацьким онлайном (MMOG) – жанр масової багатокористувацької онлайн-ігри (MMOG), що типізується такими іграми, як Neverwinter Nights, EverQuest, World of Warcraft та Galaxies Star Wars Galaxies. MMOG визначається як будь-яка гра, яка підтримує величезну кількість гравців водночас на сервері (від тисяч до сотень тисяч), як правило, всі вони грають в одному дуже великому віртуальному світі (тобто світі,

внутрішній стан якого зберігається протягом дуже тривалих періодів часу, набагато вищий за сеанс ігрового процесу будь-якого гравця). В іншому випадку досвід гри MMOG часто схожий на досвід невеликих малокористувацьких ігор. До підкатегорій цього жанру належать рольові ігри ММО (MMORPG), стратегічні ігри ММО в режимі реального часу (MMORTS) та шутери від першої особи ММО (MMOFPS).

В основі всіх MMOG – дуже потужна батарея серверів. Ці сервери підтримують стан ігрового світу, керують користувачами, які входять і виходять із гри, надають чати або послуги з передачею голосу через IP (VoIP) тощо. Майже всі MMOG вимагають, щоб користувачі платили якусь регулярну абонентську плата, щоб грати, і вони можуть робити мікро-транзакції всередині ігрового світу або поза грою. Отже, найважливіша роль центрального сервера полягає в обробці рахунків та мікро-транзакцій, які служать основним джерелом доходу розробника ігор.

Графічна здатність у MMOG майже завжди нижча, ніж ігор аналогів інших жанрів, внаслідок величезних розмірів ігрового світу та надзвичайно великої кількості користувачів онлайн, підтримуваних подібними іграми.

Це не повний перелік, оскільки існує безліч інших жанрів ігор. До прикладу: симулятори (футбол, бейсбол, футбол, гольф тощо); економічні ігри такі, як SimCity або The Sims.

1.2. Огляд ринка мобільних ігор

На початок 2021 року світовий ринок всіх ігрових додатків оцінюється у 159,3 мільярда доларів, причому 48% від цього – 77,2 мільярда доларів знаходиться в мобільних іграх. Для порівняння: в минулих роках обсяги ігрового ринку були менше ніж 152 млрд. доларів, в музичній індустрії заробили 58 млрд. доларів, а касові збори в кінотеатрах склали 42,5 млрд. доларів. Ігровий ринок, перевищує ринок звукозапису і кіноринок більше ніж на 50 млрд. доларів.

Існує декілька чинників, що призвели до такого успіху. Бізнес-модель значно змінилася за останні декілька років. Через це споживачі купують менше різних додатків, але проводять в них більше часу ніж раніше. Щоб використовувати всі можливості, розробники ігор перейшли від старої моделі разової покупки товару до нової моделі мікротранзакції. Деякі з надпопулярних можливостей для монетизації в різноманітних іграх містять продаж різноманітних персонажів, зброї, пакетів розширення контенту, ящиків з різноманітними випадковими товарами та безлічі інших предметів для більш комфортної гри.

Ігрова індустрія буда довгий час головним рушієм сегмента мобільних додатків на планеті, але пандемія коронавірусу допомогла вийти на небувалий рівень: у порівнянні з 2019 роком кількість установок додатків у 2020 збільшилася на 47%. Це на 41% вище, ніж показники зростання минулого року. На сьогодні користувачів мобільних платформ, які грають в мобільні ігри більше ніж 50%, що зрівняло ігрову категорію додатків по популярності з музичною. До прикладу, в США час, витрачений людьми на роботу з мобільними пристроями офіційно перевищив час витрачений на перегляд телебачення.

Ігри на мобільній платформі мають головну перевагу перед іншими платформами оскільки є можливість грати будь-де. Від простих повсякденних ігор до приголомшливих драйвових новинок, які швидко набирають популярність легко граються та підходять для коротких сесій у вільний час протягом цілого дня, в ігри грає майже кожен демографічний прошарок людства. Середній вік мобільних геймерів зараз становить 36,6% (у порівнянні з 28,1% у 2014 році), гендерний розкол становить 53,6% жінок, 46,4% чоловіків, а третина всіх гравців у віці 32-50 років – це далеко від традиційних стереотипів. До прикладу, у США з 2011 року ігри визнані як окремий вид мистецтва[8].

З 2021 року мобільний ринок збільшився у зв’язку з розвитком нових технологій і тенденцій. Цьому сприяли такі тенденції:

1) З розповсюдженням 5G почався бурхливий розвиток хмарних ігор. 2020 рік став роком 5G, що призвело до початку глобального впровадження в усіх країнах світу даної технології. Комерційне впровадження 5G почалося у 2020, а широкомасштабні рухи продовжаться до 2025 року.
2) Хмарні сервіси дають більше можливостей для користувачів мобільних платформ для мобільних ігор. У 2020 році компанії Google і Microsoft випустили свої нові хмарні ігрові рішення. Віце-президент з маркетингу в компанії IronSource, описує це: «В цілому малоймовірно, що за короткий час хмарні гри замінять консолі й ПК або будь-яким чином вплинуть на аудиторію казуальних гравців, що може перетворити їх на фанатів хмарних рішень. З мого досвіду, всі нові формати та технології які з'являються на ринку мобільних ігор не перевернуть весь ринок кардинально, але створять декілька нових прошарків. Розширюючи діапазон можливого ігрового досвіду нових та старих геймерів і створюючи більший доступ людям до традиційних ігор, хмарні рішення, ймовірно внесуть значний розвиток всього ринку ігор».
3) Казуальні геймери не будуть купувати щомісячну підписку на хмарні сервіси, що не скажеш про Хард-корних геймерів Компанії Apple і Google, які на весь світ оголосили про свої нові сервіси, які працюють по новій системі платних підписок. Компанії Apple і Google пропонують гравцям підписку на нові ігрові сервіси – змінюючи свій дохід від ігрових покупок і реклами в додатках на щомісячну плату з гравця. Проте, не дивлячись на сформовану тенденцію монетизації компаніями, ми все ще маємо безліч невідомих факторів для повноцінного аналізу та прогнозів на майбутнє: ні компанія Apple, ні компанія Google, ні сторонні розробники, що працюють з компаніями, не пояснили повноцінно як буде працювати бізнес-модель даного напрямку в ігровій сфері.

Висновки до розділу 1. Отже, проаналізовано ринок ігрової індустрії, здійснено класифікацію комп'ютерних ігор за жанрами та їх порівняння, а

також найпопулярніші типи ігрових платформ. Мобільні ігри зазнали серйозних змін завдяки розробці і розвитку нових технологій, таких як Інтернет, ігрові рушії та ін. Також зросла кількість мобільних ігрових додатків, за допомогою яких мільйони користувачів мають можливість витрачати вільний час для розваг. В наш час мобільні технології та різні мобільні ігри є перспективною сферою нових ідей. Через це розробка мобільних додатків є актуальною.

РОЗДІЛ 2. АНАЛІЗ ЗАСОБІВ ДЛЯ РОЗРОБКИ 3D ВІДЕОІГОР

2.1. Визначення ігрового рушія

Ігровий рушій − середовище розробки програмного забезпечення, яке призначене для створення відеоігор. Сьогодні, ігрові рушії значно спрошують завдання з розробки ігор завдяки існуючим шаблонам та ассетам, які можна використовувати повторно. Тим самим мінімізуючи, а в деяких випадках навіть повністю виключаючи необхідність глибоких знань та навичок програмування. Також вони дають можливість експортувати проект на велику кількість ігрових платформ, включаючи мобільні пристрої з мінімальними змінами.

Термін «Ігровий рушій» вперше з’явився у середині 1990-х років у відношенні до шутерів від першої особи (FPS), таких як Doom від компанії id Soft. Doom був розроблений з досить чітко визначеним поділом між основними компонентами програмного забезпечення (наприклад, тривимірна система візуалізації графіки, система виявлення зіткнень або аудіосистема) та об’єктами, ігровими світами та правилами гри, які включали в себе ігровий досвід гравця. Значення цього розмежування стало очевидним, коли розробники почали ліцензувати ігри та перевлаштовувати їх у нові продукти, створюючи нові ігрові об’єкти, карту світу, зброю, персонажів, транспортні засоби та правила гри, лише зі змінами програмного забезпечення. Це ознаменувало народження спільноти розробників модифікацій – групи окремих геймерів та невеликих незалежних студій, які будували нові ігри, модифікуючи існуючі ігри, використовуючи безкоштовні набори інструментів, надані оригінальними розробниками.

Наприкінці 1990-х років деякі ігри, такі як Quake III Arena та Unreal, були розроблені з урахуванням повторного використання та «відхилення». Рушії були налаштовані за допомогою мов скриптів, таких як Id Quake C, і ліцензування двигунів стало додатковим потоком доходів для розробників, які їх створили. Сьогодні розробники ігор можуть ліцензувати ігровий

движок та повторно використовувати значні частини своїх основних компонентів програмного забезпечення для створення ігор. Хоча ця практика все ще передбачає значні інвестиції в інженерну розробку програмного забезпечення, вона може бути набагато економічною, ніж розробка всіх основних компонентів рушія.

Межа між грою та її двигуном часто розмита. Деякі двигуни досить чітко розрізняють цю межу, в той час як інші майже не мають змоги розділити ці два поняття. В одній грі код візуалізації може спеціально «знати», як намалювати один певний об’єкт. В іншій грі механізм візуалізації може надавати матеріали загального призначення та засоби для затінення, що дозволить створити безліч різноманітних об’єктів.

Жодна студія не робить абсолютно чіткого розмежування між грою та двигуном, що зрозуміло, враховуючи, що визначення цих двох компонентів часто змінюються в міру того, як визначений дизайн гри. Можливо, архітектура, керування даними – це те, що відрізняє ігровий движок від програмного забезпечення, яке є грою, але не двигуном. Коли гра містить жорстко закодовану логіку, чи правила гри або використовує спеціальний код для візуалізації конкретних типів ігрових об’єктів, використовувати це програмне забезпечення для створення іншої гри стає важким або неможливим.

Отже, термін «ігровий рушій» можна визначити, як програмне забезпечення, що розширюється і може використовуватися як основа для багатьох різних ігор без великих змін.

2.2. Аналіз та порівняльна характеристика ігрових рушіїв

Godot − це 2D та 3D, крос-платформний, безкоштовний і відкритий ігровий рушій, який дозволяє розробляти відеоігри, орієнтовані на ПК, мобільні та веб-платформи.

Архітектура рушія побудована навколо концепції "вузлів". Вузли організовані всередині "сцен", які є групами вузлів. Усі ігрові ресурси, включаючи скрипти та графічні ресурси, зберігаються як частина файлової системи комп'ютера, а не в базі даних. Таке рішення для зберігання даних полегшує співпрацю між групами розробників відеоігор, які використовують системи контролю версій програмного забезпечення.

Godot підтримує такі мови програмування як: C++, C# та будь-які інші мови з прив'язками GDNative, такі як Rust, Nim та D. Рушій має також свою скриптову мову GDScript, яка дуже схожа на Python, а також підтримує візуальне кодування за допомогою власної мови візуального програмування VisualScript.

Графічний рушій Godot використовує OpenGL ES 3.0 або OpenGL ES 2.0. Він підтримує: динамічні тіні, карти нормалей, дзеркальність, як запечене так і динамічне глобальне освітлення, а також різні ефекти постобробки. Godot також включає в себе окремий 2D-графічний рушій, який працює незалежно від 3D-рушія. Також він містить систему анімацій з інтерфейсом для скелетної анімації, анімаційних дерев та анімацій змішування. Практично будь-яка змінна, що створена в ігровому об'єкті, може бути анімована.

Можна також виділити декілька недоліків Godot:

− недостатня кількість документації;

− проблеми з імпортом 3D-ресурсів;

− слабкий фізичний рушій.

Щодо філософії архітектури Godot [9], то Godot надає свої власні інструменти для задоволення найбільш поширених потреб. Він має вбудований редактор коду, редактор анімації, редактор tilemap, редактор шейдерів, зневаджувач, профілювальник, можливість перезавантаження на ходу локально і на віддалених пристроях і багато іншого.

Мета полягає в тому, щоб запропонувати повний пакет для створення ігор і безперервного навчання. Є можливість працювати з зовнішніми

програмами, якщо для них є плагін імпорту. Або можна створити його, наприклад, Tiled Map Importer.

Частково тому Godot пропонує власні мови програмування GDscript та VisualScript разом із C#. Вони розроблені для потреб розробників ігор та дизайнерів ігор, і вони тісно інтегровані в рушій та редактор.

GDScript дозволяє писати код із використанням синтаксису, заснованого на відступах, але він виявляє типи та пропонує якість автоматичного завершення статичної мови. Він також оптимізований для ігрового коду з вбудованими типами, такими як вектори та кольори.

Зауважимо, що за допомогою GDNative є можливість писати високоефективний код, використовуючи компільовані мови, такі як C, C++, Rust, або Python (використовуючи компілятор Cython), без перекомпіляції рушія.

Суттєвим є також, що Godot пропонує повністю відкритий вихідний код за ліцензією MIT. Це означає, що всі технології, які постачаються разом із ним, також мають бути безкоштовними (як і вільними). Здебільшого вони розробляються з нуля учасниками.

Будь-хто може підключити пропрієтарні інструменти для потреб своїх проектів – вони просто не будуть поставлятися разом із рушієм. Наприклад, Google AdMob або FMOD. Натомість будь-який з них може бути підключений як зовнішній плагін.

З іншого боку, відкритий вихідний код означає, що є можливість *навчити і розширити редактор* за бажанням. Ви також можете легко зневаджувати ігри, оскільки Godot друкує помилки із трасуванням стека, навіть якщо вони надходять із самого рушія.

Редактор Godot – це гра Godot [9]. Це означає, що редактор Godot працює на ігровому рушії. Він використовує власну систему інтерфейсу рушія, може перезавантажувати код і сцени під час тестування проектів, або запускати ігровий код у редакторі. Це означає, що є можливість

використовувати *той самий код* і сцени для своїх ігор, або *створювати плагіни та розширювати редактор*.

Це призводить до надійної та гнучкої системи користувацького інтерфейсу, оскільки вона приводить в дію сам редактор. За допомогою ключового слова tool можна запустити будь-який ігровий код у редакторі. Для цього достатньо помістити ключове слово tool на початку будь-якого файлу GDScript, і він запуститься в редакторі. Це дозволяє імпортувати та експортувати плагіни, створювати плагіни, такі як редактори користувацького рівня, або створювати крипти з тими ж вузлами та API, які ми використовуємо у своїх проектах.

Godot пропонує спеціалізовані рушії 2D та 3D візуалізації. В результаті базовою одиницею для 2D сцен є пікселі. Незважаючи на те, що рушії є окремими, ми можемо візуалізувати 2D в 3D, 3D в 2D і накладати 2D спрайти та інтерфейси поверх створеного 3D-світу.

Godot створений його спільнотою, для спільноти та для всіх творців ігор. Саме основні потреби користувачів та відкриті дискусії визначають основні оновлення цього ігрового рушія. Нові функції основних розробників часто зосереджуються на тому, що в першу чергу принесе користь користувачам.

Кожен ігровий рушій відрізняється і відповідає різним потребам. Вони не тільки пропонують ряд функцій, але й дизайн кожного рушія є унікальним. Це призводить до різних робочих процесів і різних способів формування структур відеоігор.

В основі Godot лежить об’єктно-орієнтований дизайн із гнучкою системою сцен та ієрархією вузлів. Він не містить строгих шаблонів програмування для того, щоб запропонувати інтуїтивно зрозумілий спосіб структурування створеної гри.

В Godot можна складати або об’єднувати сцени. Сцена може бути персонажем, предметом, рівнем, частиною рівня, чим завгодно. Він працює як клас у чистому коді, за винятком того, що є можливість створити його за

допомогою редактора, використовуючи лише код або змішуючи та поєднуючи обидва. Це відрізняється від збірних елементів, які можна знайти в кількох 3D-движках, оскільки є можливість успадковувати та розширювати ці сцени. Це допомагає будувати проекти так, щоб їх структура відповідала дизайну гри.

Також Godot пропонує багато різних типів об’єктів, які називаються вузлами, кожен з яких має певне призначення. Вузли є частиною дерева і завжди успадковують від своїх батьків аж до класу Node. Незважаючи на те, що механізм містить деякі вузли, як-от форми зіткнень, які використовуватиме батьківське фізичне тіло, більшість вузлів працюють незалежно один від одного.

Коли запускається Godot, перше вікно, яке відкривається, – це менеджер проектів. На стандартній вкладці «Проекти» можна керувати існуючими проектами, імпортувати або створювати нові тощо.

У верхній частині вікна є ще одна вкладка під назвою «Проекти бібліотеки ресурсів». У бібліотеці активів з відкритим кодом можна шукати демонстраційні проекти, шаблони та завершені проекти, включно з багатьма проектами, розробленими спільнотою. Також можна змінити мову редактора за допомогою спадного меню праворуч від версії движка у верхньому правому куті вікна. За замовчуванням це англійська (EN).

Коли відкривається новий або існуючий проект, відкривається інтерфейс редактора (рисунок 2.1). Розглянемо його основні напрямки.

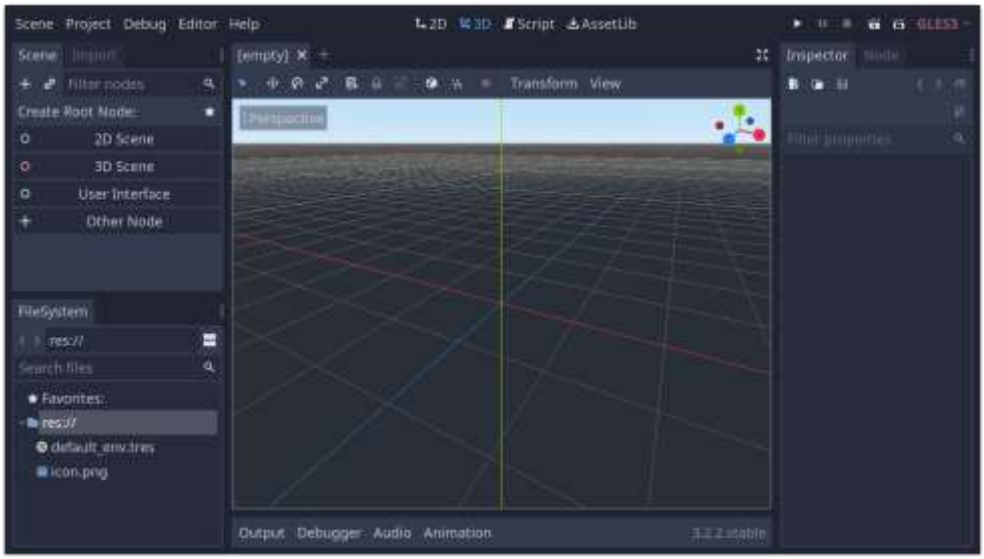


Рис. 2.1. Інтерфейс редактора Godot

За замовчуванням він містить меню, головні екрани та кнопки тестування відтворення вздовж верхнього краю вікна. У центрі розташоване вікно перегляду з панеллю інструментів угорі, де знаходяться інструменти для переміщення, масштабування або блокування вузлів сцени.

По обидві сторони вікна огляду розташовані доки. Внизу вікна розташована нижня панель. Панель інструментів змінюється залежно від контексту та вибраного вузла.

FileSystem містить список файлів проекту, будь то сценарії, зображення, зразки аудіо тощо. «Сцена» містить список вузлів активної сцени. Інспектор дозволяє редагувати властивості вибраного вузла. Нижня панель, що розташована під вікном перегляду, є хостом для консолі налагодження, редактора анімації, звукового мікшера, тощо. Вони займають дорогоцінний простір, тому за замовчуванням вони згорнуті. При виборі вузла, він розгортається вертикально. Нижче знаходиться редактор анімації.

На 3D-екрані можна працювати з сітками, світлом і рівнями дизайну для 3D-ігор. Натиснувши кнопку перспективи під панеллю інструментів, відкриється список параметрів, пов’язаних із 3D-видом.

Шукати інформацію про клас, метод, властивість, константу або сигнал можна будь-яким із наведених нижче методів:

– Натискання F1(або в macOS) будь-де в редакторі Alt + Space;

- «Пошук у довідці» у верхньому правому куті головного екрана сценарію.
- кнопкою меню «Довідка» та «Пошук довідки»;
- Ctrl клавіші на назві класу, назві функції або вбудованій змінній у редакторі сценаріїв.

Godot надає власні інструменти для найбільш поширених потреб. Він має спеціальну робочу область для створення сценаріїв, редактор анімації, редактор карти фрагментів, редактор шейдерів, можливість перезавантаження локально та на віддалених пристроях тощо. Також можна використовувати його для перегляду доступних об'єктів і методів.

Мета полягає в тому, щоб запропонувати повний пакет для створення ігор і безперервний досвід користувача. Також можнае працювати із зовнішніми програмами, якщо для них є плагін імпорту. Або можна створити його, як, наприклад, Імпортер мозаїчної карти.

Частково тому Godot пропонує власні мови програмування GDScript і VisualScript разом із C#. Вони створені для потреб розробників і дизайнерів ігор і тісно інтегровані в механізм і редактор.

GDScript дозволяє писати код, використовуючи синтаксис на основі відступів, але він визначає типи та пропонує якість автоматичного завершення статичної мови. Він також оптимізований для ігрового коду з вбудованими типами, такими як вектори та кольори.

VisualScript – це мова програмування на основі вузлів, яка добре інтегрується в редактор. Ви можете перетягувати вузли або ресурси в граф, щоб створювати нові блоки коду.

Робоче середовище 3D не містить стільки інструментів, скільки робоче середовище 2D. Знадобляться зовнішні програми або доповнення для редагування рельєфу, анімації складних персонажів і так далі. Godot надає повний API для розширення функціональності редактора за допомогою коду гри [2].

Unity – це крос-платформенний ігровий рушій, що розроблений компанією Unity Technologies. Вперше його було представлено в 2005 році на конференції Apple. Тоді він призначався тільки для розробки в OS X. На сьогоднішній день Unity є одним з найпопулярніших ігрових рушіїв. Наприклад, більше половини всіх нових мобільних ігор розробляються на Unity, а також більше 70% ігор віртуальної да доповненої реальності. Unity підтримується на таких платформах як Windows, macOS і Linux, а сам рушій на сьогоднішній день дозволяє створювати ігри для більш ніж 25 різних платформ, включаючи мобільні пристрої, ПК, консолі, та віртуальну реальність. Рушій дає можливість розробникам створювати відеоігри як в 2D, так і в 3D.

Unity надає основний API сценаріїв на C# як для редактора Unity у формі плагінів, так і для самих відеоігор, а також забезпечує просте «drag and drop» середовище. В нових версіях Unity, стала доступною мова візуальних сценаріїв Bolt, яка дозволяє створювати ігрові механіки та інтерактивні системи без написання єдиного рядка коду (раніше Bolt була додатковим платним розширенням). Такі ресурси, як Bolt для «візуальних сценаріїв» (visual scripting), дозволяють розробникам впроваджувати ігрову логіку у свої проекти, використовуючи інтуїтивно зрозумілі графічні елементи без будь-яких знань про код або IDE.

У 2D-іграх Unity дозволяє імпортувати спрайти та має вдосконалений 2D рендеринг. Для 3D-ігор Unity дає можливість визначити стиснення текстур, mip-мапи та налаштування роздільної здатності для кожної ігрової платформи, яку підтримує рушій, забезпечує підтримку рельєфного текстурування, відображення, паралакс, screen space ambient occlusion (SSAO), динамічні тіні з використанням карт тіней, ефекти рендерингу та ефекти пост обробки зображення.

Важливим плюсом Unity є Unity Asset Store. Він дозволяє розробникам продавати або ділитися своїми ассетами. Магазин був запущений у 2010 році, до 2018 року через нього було завантажено близько 40 мільйонів ігрових

ассетів. Це допомагає одним розробникам зекономити час розробки своїх проектів, а іншим заробити. Unity Asset Store надає готові сценарії та інструменти для розробки загальних функцій, таких як контролер від першої особи або система «інвентар» [3].

Щодо програмування. Хоча програмування – це корисна навичка для розробки проектів зі складною інтерактивністю в Unity, не обов'язково бути програмістом, щоб створювати проекти за допомогою Unity. Наприклад:

- деякі типи проектів, наприклад 3D-візуалізації та анімації, взагалі не потребують коду;
- такі ресурси, як Bolt для «візуальних сценаріїв» (visual scripting), дозволяють розробникам впроваджувати ігрову логіку у свої проекти, використовуючи інтуїтивно зрозумілі графічні елементи без будь-яких знань про код або IDE;
- Unity Asset Store надає готові сценарії та інструменти для розробки загальних функцій, таких як контролер від першої особи або система «інвентар»;
- використовуючи Google у поєднанні з такими сайтами, як Unity Answers, Unity Forums, і Stack Overflow, розробники можуть копіювати, вставляти та змінювати код, наданий іншими розробниками.

Головним мінусом Unity є те, що рушій значно відстає з графічної точки зору. Він не пропонує безліч інструментів для створення складної графіки, на відміну від інших ігрових рушіїв. Також проекти, що розроблені на Unity, споживають більше пам'яті, а через це в свою чергу, виникає багато помилок OOM та проблем налагодження в додатках.

Unity має чотири версії: Personal, Plus, Pro та Enterprise. Для даного проекту оптимальною є перша версія, яка є безкоштовною, за умовою, що дохід від проекту не перевищує 100 000$ на рік. Інші версії коштують від $399 до $1,800 на рік.

Здійснену порівняльну характеристику функціональних можливостей ігрових рушіїв Godot та Unity подамо у вигляді таблиці 2.1

Таблиця 2.1

Порівняльна характеристика ігрових рушії Godot та Unity

	Unity	**Godot**
Open Source	Ні	Так
Мова програмування	C#	GodotScript (C#)
Системні вимоги	Потрібно середні характеристики ПК	Запускається на простих і слабких ПК
Платформи	PC, консолі, мобільні платформи	PC, мобільні платформи
Ліцензія	Для персонального використання безкоштовно, поки продажі гри не перевищують 100000$	Повністю безкоштовний
Простота встановлення	Так	Так
Переваги	Велика кількість мов інтерфейсу	Велика кількість мов інтерфейсу
	Простий в освоєнні	Простий в освоєнні
	Багато уроків і документації	Безкоштовний
	Багато вакансій на ринку праці	Постійно розвивається
	Великий вибір готових 3D моделей	Підходить новачкам
	Постійно розвивається, підходить новачкам, велике ком'юніті	Мало займає місця на ПК
Недоліки	Повільна робота при роботі з великими	Мало уроків і документації

	сценами	
	Займає багато місця на ПК	Відсутня документація українською мовою
	Закритий код рушія	Має менше технологій чим Unity
	Оптимізація проектів	Відсутність вакансії на ринку праці
	Не повністю безкоштовний	Українське ком'юніті не багаточисельне

2.3 Засоби для розробки 3D ігрових додатків

Blender – це безкоштовне програмне забезпечення для створення та редагування тривимірної графіки. Завдяки кросплатформеності, відкритого вихідного коду, доступності та функціональності пакет отримав заслужену популярність не тільки серед новачків, але й серед просунутих 3D-моделерів. За мірою розвитку програми її вибирають в якості робочого інструменту для більш серйозних проектів, що не є дивним. По суті, це додаток практично не поступається за кількістю можливостей і функціональності більш просунутими пакетами 3D-графіки. І при цьому все безкоштовно.

В представленій роботі Blender використовується для створення 3D-моделей. Після чого вони експортуються в ігровий рушій.

Програма є чудовою підмогою для знайомства з 3D графікою та функціонуванням базових інструментів створення та редагування 3D об'єктів, адже Blender поєднує в собі набір опцій, які окремо зустрічаються у професійних тривимірних редакторах. Можна нехтувати тим, що в ньому зібрано потроху від кожної відомої програми для створення 3D моделей. Але в той же час це – повністю самостійний, унікальний пакет тривимірної графіки, не схожий на жодну іншу програму.

На сьогоднішній день це повноцінний 3D редактор, в якому користувача зустрічає інтерфейс, що повністю програмується, і унікальна внутрішня файлова система. Оболонка програми на перший погляд може здатися незручною та незрозумілою, але після настроювання гарячих клавіш працювати в Blender стає просто та зручно. Як мову програмування додаток використовує Python. Володіючи цією мовою програмування, можна створювати власні інструменти, редагувати інтерфейс і сам принцип роботи програми. Приємним бонусом є доступність пакета на різних операційних системах з Windows, GNU/Linux та Mac OSX.

Blender – програмний пакет для створення тривимірної комп'ютерної графіки, що включає засоби моделювання, анімації, рендерінгу, після-обробки відео. До версії 2.80 містив рушій Blender Game Engine для створення відеоігор. Пакет є вільним програмним забезпеченням та розповсюджується під ліцензією GNU GPL.

Особливостями пакету є малий розмір, висока швидкість рендерінгу, наявність версій для багатьох операційних систем – FreeBSD, GNU/Linux, Mac OS X, SGI Irix 6.5, Sun Solaris 2.8 (sparc), Microsoft Windows, MorphOS та Pocket PC. Пакет має такі функції, як симуляція динаміки твердих тіл (Rigid Body), рідин (Liquid simulation) та м'яких тіл (Soft body), редагування матеріалів і геометрії за принципом вузлів (Nodes), велику кількість легко доступних розширень, написаних мовою Python.

Характерними особливостями пакету Blender є його невеликий розмір та підтримка багатьох популярних операційних систем. Він підтримує роботу з багатьма геометричними примітивами – базовими полігональними моделями (куб, сфера, циліндр тощо), кривими Безьє, поверхнями NURBS, metaballs, векторними шрифтами. Шляхом їх перетину та зміни розташування й розмірів окремих полігонів створюються всі інші, складніші, об'єкти. Є функція малювання довільних кривих нарисним олівцем (Grease Pencil).

У Blender будь-яка сутність, з якою взаємодіє користувач, називається об'єктом. Це може бути як полігональна модель чи крива, так і джерело світла, камера огляду, арматура моделі тощо, котрі видимі при редагуванні, але не відображаються в фінальній роботі. При цьому дані об'єкта (певна форма/функція) відділені від нього, тому декілька об'єктів здатні використовувати одні й ті ж дані.

Основу інтерфейсу складають горизонтальні вкладки (робочі простори), кожна з яких відведена під певну категорію функцій, що дозволяє легко перемикатися між різними завданнями, забезпечуючи різні дії над 3D моделями в одному вікні. Праворуч у кожній вкладці містяться панелі інструментів, які мають власні вкладки, розташовані вертикально. Практично кожна функція має відповідне їй поєднання клавіш, і враховуючи кількість наданих можливостей у Blender, кожна клавіша включена в більш ніж одне поєднання (shortcut). З того часу як Blender став проектом з відкритим вихідним кодом, було додано повні контекстні меню до усіх функцій, а використання інструментів зроблене логічнішим та гнучкішим. Користувацький інтерфейс підтримує колірні схеми оформлення, прозорі плаваючі елементи, які розширюють функціональність Blender-а. До окремих об'єктів і навіть їхніх полігонів можна прикріплювати нотатки.

Користувацький інтерфейс Blender-а містить такі головні вкладки:

- Layout (Подання) – головний робочий простір для перегляду сцени та її простих налаштувань.
- Modelling (Моделювання) – модифікація геометрії інструментами моделювання.
- Sculpting (Скульптинг) – модифікація геометрії інструментами скульптингу.
- UV Editing (Редагування розгортки) – підлаштування координат текстури до поверхні тривимірної моделі.
- Texture Paint (Малювання текстури) – малювання текстур на поверхні моделі.

- Shading (Шейдинг) – надання моделі матеріалів.
- Animation (Анімація) – налаштування зміни моделі з часом.
- Rendering (Рендеринг) – налаштування, показ і аналіз фінальної картинки.
- Compositing (Компонування) – комбінування картинок і їхня пост-обробка.
- Geometry Nodes (Геометричні вузли) – процедурне моделювання.
- Scripting (Скриптування) – написання скриптів для автоматизації дій.

Для рендерингу Blender використовує декілька рушіїв. Рендеринг може відбуватися як на центральному, так і на графічному процесорі. Вигляд відрендереної картинки залежить від текстур, матеріалів, освітлення, обраної камери огляду та налаштувань обраного рушія. Об'єкти можуть поміщуватися на різні шари рендерингу, яким відповідають певні колекції об'єктів. Це потрібно, наприклад, для симуляції розмиття в міру віддалення від камери.

- Workbench – призначений для дуже швидкого відображення моделей з урахуванням лише кольору та освітлення.
- Eevee – дозволяє швидко отримувати картинку, проте без складних ефектів, як-от відбиття чи заломлення променів.
- Cycles – містить фізичний рушій, здатний обчислювати трасування променів. Завдяки цьому Cycles під силу відтворювати фотореалістичну картинку, проте на це потрібно більше час.

Також на безоплатній основі доступні зовнішні рушії, такі як:

- Octane Render від Otoy з можливістю безкоштовного використання на одній графічній карті;
- LuxCoreRender;
- ProRender від Radeon.

Системні вимоги Blender подано у таблиці 2.2.

Таблиця 2.2

Системні вимоги Blender

Обладнання	**Мінімальні**	**Рекомендовані**	**Оптимальні**
Процесор	64-бітний двоядерний 2Ghz з підтримкою SSE2	64-бітний чотирьохядерний	64-бітний восьмиядерний
Оперативна пам'ять	4 GB	16 GB	32 GB
Відеокарта	OpenGL 3.3 карта з 1 ГБ відеопам'яті	OpenGL 3.3 карта з 4 ГБ відеопам'яті	OpenGL 3.3 карта з 12+ ГБ відеопам'яті
Екран	1280×768 пікселів, 16-бітний колір	1920×1080 пікселів, 24-бітний колір	декілька, 1920×1080 пікселів, 24-бітний колір
Прилади для вводу	Мишка з трьома кнопками	Мишка з трьома кнопками	Мишка з трьома кнопками та графічний планшет

І хоча для повноцінної роботи потрібна мишка з трьома кнопками або коліщатком-кнопкою, програма може емулювати її, використовуючи гарячі клавіші.

Робота з тривимірними моделями відбувається у сцені, розкресленій координатною сіткою. Об'єкти сцени об'єднуються в так звані колекції. За промовчуванням кожна сцена має одну колекцію, але користувачі вільні створювати нові колекції та переміщувати між ними об'єкти для згрупування своєї роботи. Схема сцени відображається за промовчуванням праворуч вгорі. До сцени може додаватися фон або площина із зображенням-зразком для моделювання. Об'єкти можуть бути неактивними (з ними не відбувається взаємодії), активними (відбувається непряма взаємодія) та вибраними (користувач взаємодіє конкретно з цим об'єктом). Всі вони мають координати походження, що враховуються при переміщенні та деформації, та виходять з поворотної точки, що може перебувати за межами самого об'єкта. Blender містить інструменти для моделювання методом скульптингу, що симулює ліплення з глини.

На полігональні моделі можуть накладатися текстури та матеріали. Текстура визначає вигляд моделі (колір, прозорість, імітація шорсткості,

блиску тощо), а матеріал додатково визначає взаємодію з іншими об'єктами (заломлення променів світла, відображення, свічення). Створення текстур і матеріалів відбувається за допомогою наочних схем, які складаються з вузлів (нодів, nodes) і зв'язків між ними. Існують вузли, що задають масштаб, яскравість, змішування текстур і т. д. Текстури й матеріали можуть генеруватися процедурно.

Blender містить інструменти анімації, серед яких inverse kinematics, арматурна (скелетна) та сіткова деформація, анімація за ключовими кадрами, нелінійна анімація, timeline, vertex weighting, constraints, динаміка м'яких тіл, включаючи визначення колізій форми об'єктів при взаємодії, динаміка рідин, Bullet динаміка твердих тіл, система волосся на основі частинок та система частинок при визначенні колізій об'єктів.

Програмний пакет можна розширювати доповненнями, котрі упаковуються в архів .zip або являють собою файли .py (Python). Такі доповнення дозволяють, наприклад, генерувати складні моделі, додавати нові інструменти. Python використовується також як засіб імпорту/експорту файлів (наприклад COLLADA), автоматизації завдань.

Висновки до розділу 2. Отже, швидке зростання мобільних ігор у секторі розваг стало поштовхом для створення спеціальних програм, а саме ігрових рушіїв. Ці технології розроблені для полегшення роботи розробників комп'ютерних та мобільних ігрових додатків, оскільки більшість питань, пов’язаних із базовими потребами створення ігор, вирішено у самому рушії та дозволяє програмісту зосередитися на розробці ігрової механіки, а не взаємодії з апаратним забезпеченням. Загалом розглянуті ігрові рушії є одними з найпопулярніших рушіїв в ігровій індустрії на сьогодні. Вони мають багато корисних інструментів, легких в освоєнні, основні з яких проаналізовано.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРАКТИЧНОЇ ЧАСТИНИ

3.1. Створення 3D моделей

Для майбутньої відео-гри в жанрі аркада потрібно зробити 3D-моделі, які будуть імпортуватися і в подальшому використовуватися в проектах. Відкриваємо Blender. Натискаємо в верхньому лівому куті пункт «Add» зі списку обираємо «Cube» (рисунок 3.1).

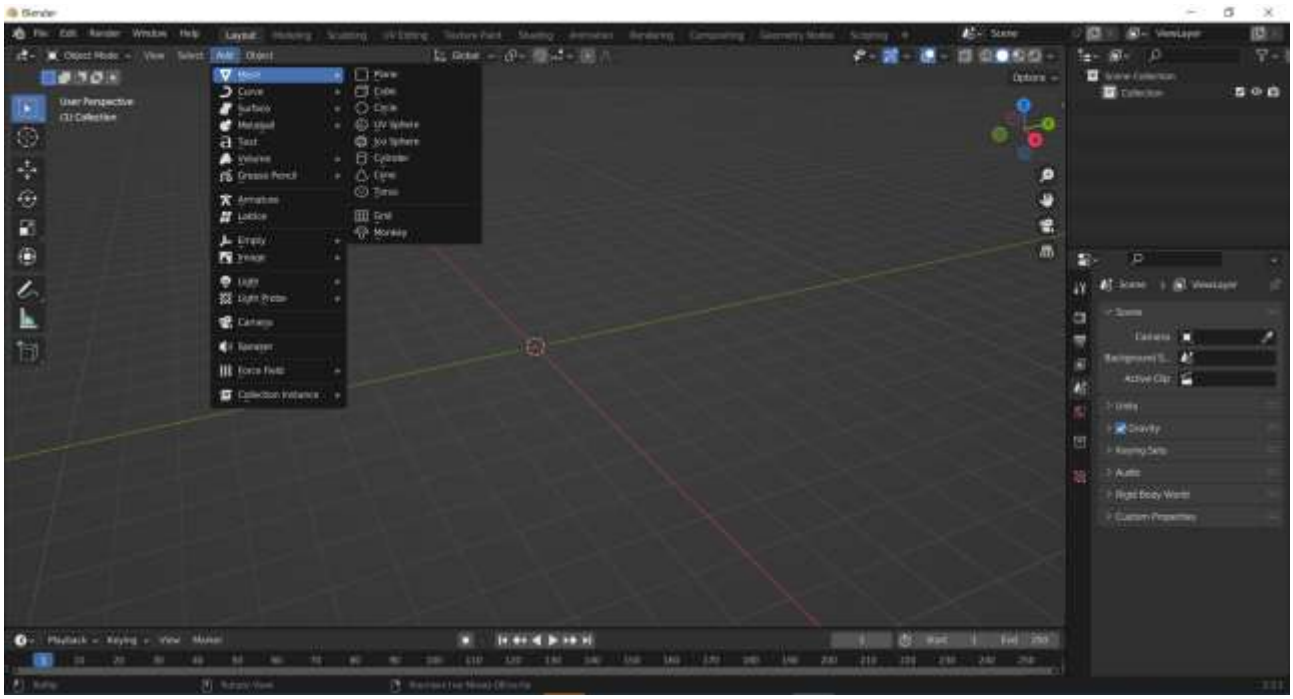


Рис. 3.1. Додавання 3D моделі

В «Viewport» з’являється 3D-модель куба, після цього натискаємо клавішу «Tab», щоб перейти в режим редагування. Потім натискаємо клавішу «s» та зменшуємо куб по осі «z» (рисунок 3.2). В результаті отримуємо платформу, яка буде використовуватися для побудови маршруту, який гравець має пройти.

Аналогічно створюємо модель гравця. В нашому випадку – це буде сфера, оскільки примітивна 3D графіка добре підходить для мобільних проектів, щоб не навантажувати пристрій і охопити більшу кількість гравців. При додаванні сфери зменшуємо кількість вершин. Це краще вплине на оптимізацію проекту (рисунок 3.3).

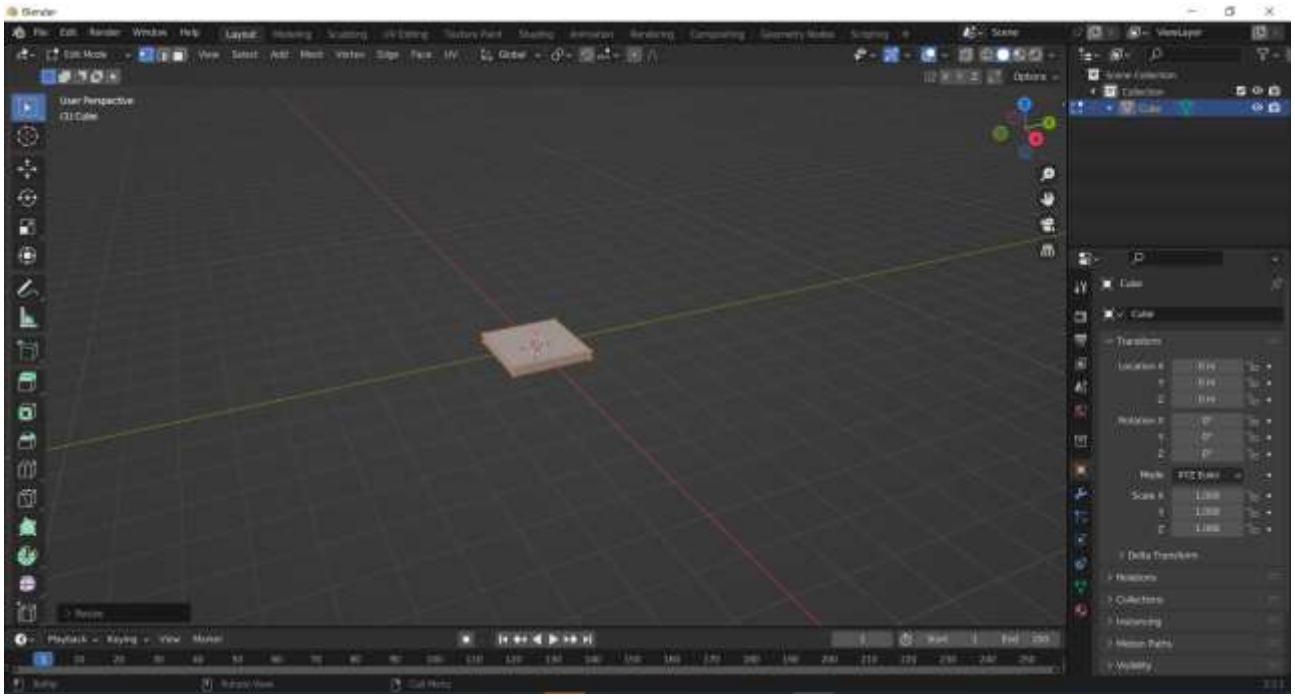


Рис. 3.2. Платформа для відеогри

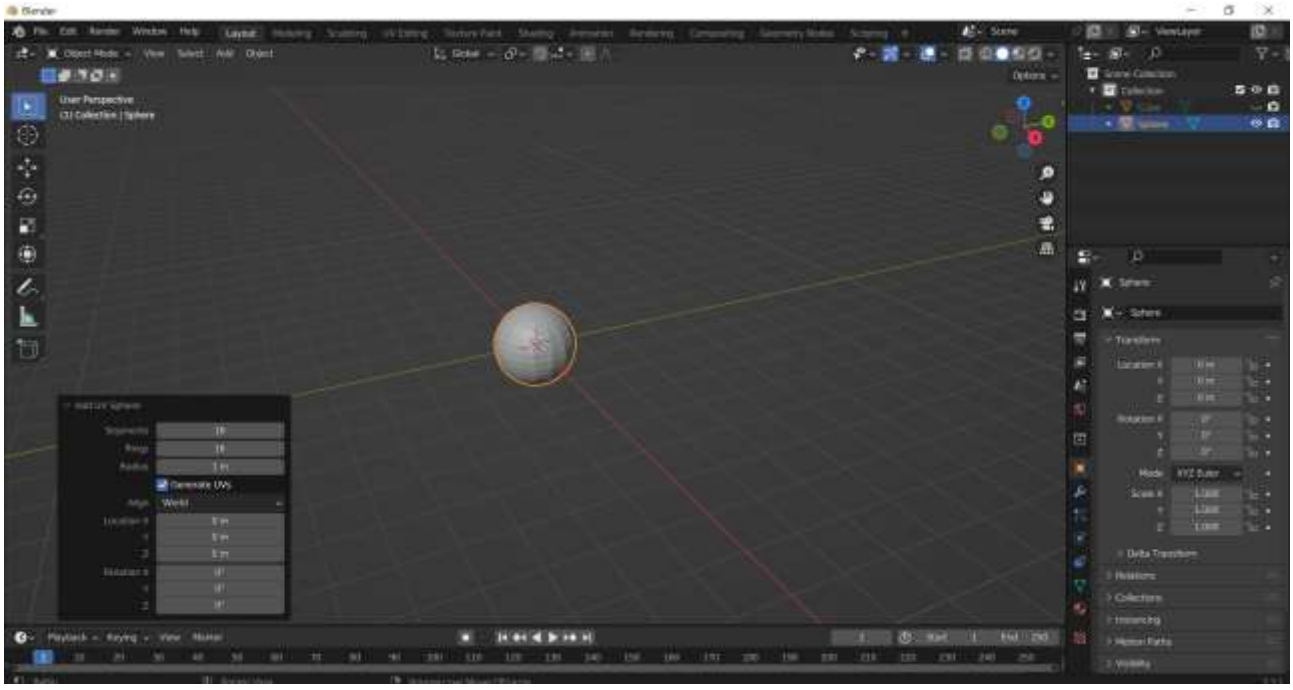


Рис. 3.3. Створення сфери

Для того, щоб сфера не втрачала свою якість, використовуємо Shade Smooth, натиснувши праву кнопку миші. Це допомагає згладжувати нормалі

моделі. В результаті отримуємо просто оптимізовану модель гравця (рисунок 3.4).

Рис. 3.4. Модель гравця

Створюємо модель прапора. При взаємодії гравця з ним, буде завершуватися рівень і здійснюватися перехід на інший рівень. Додаємо об’єкт куб, видаляємо верхній полігон, натиснувши на клавішу «X». Після цього зменшуємо модель по осі «z», виділяємо верхні вершини і використовуємо інструмент «Exctrude» і за допомогою нього створюємо модель (рисунок 3.5).

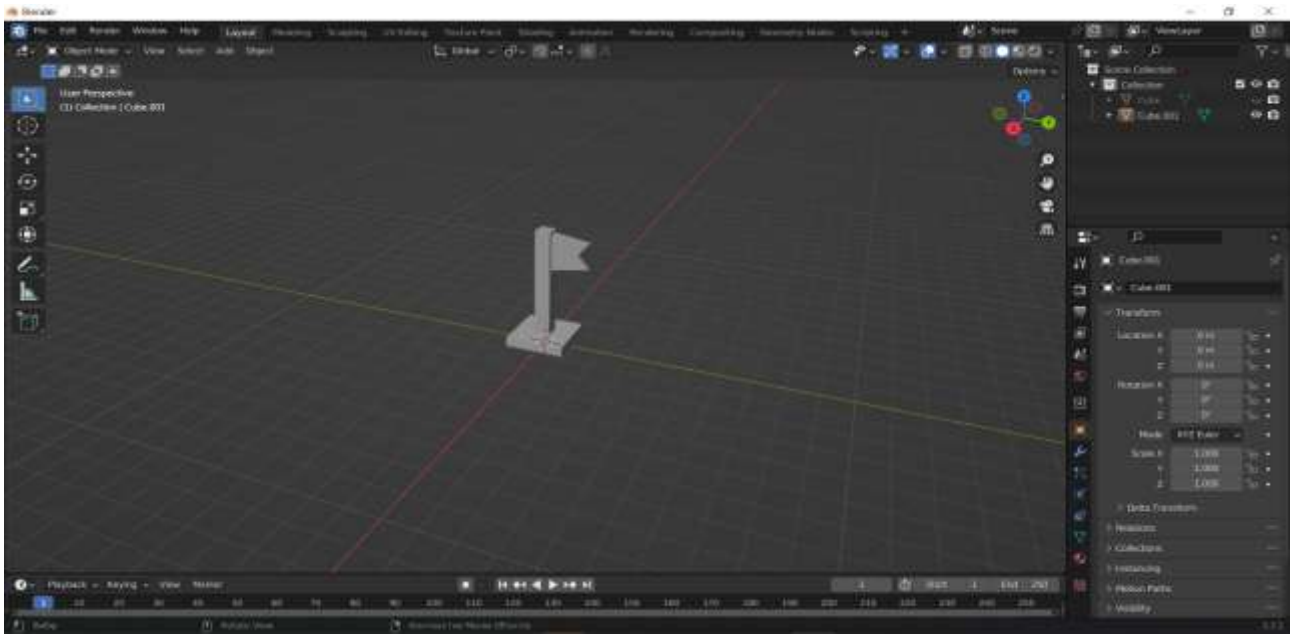


Рис. 3.5. Модель прапора

Для експортування моделей в ігровий рушій Godot Engine, обираємо потрібну 3D-модель – пункт «File», обираємо формат glTF 2.0 і обираємо папку куди потрібно експортувати модель (рисунок 3.6).

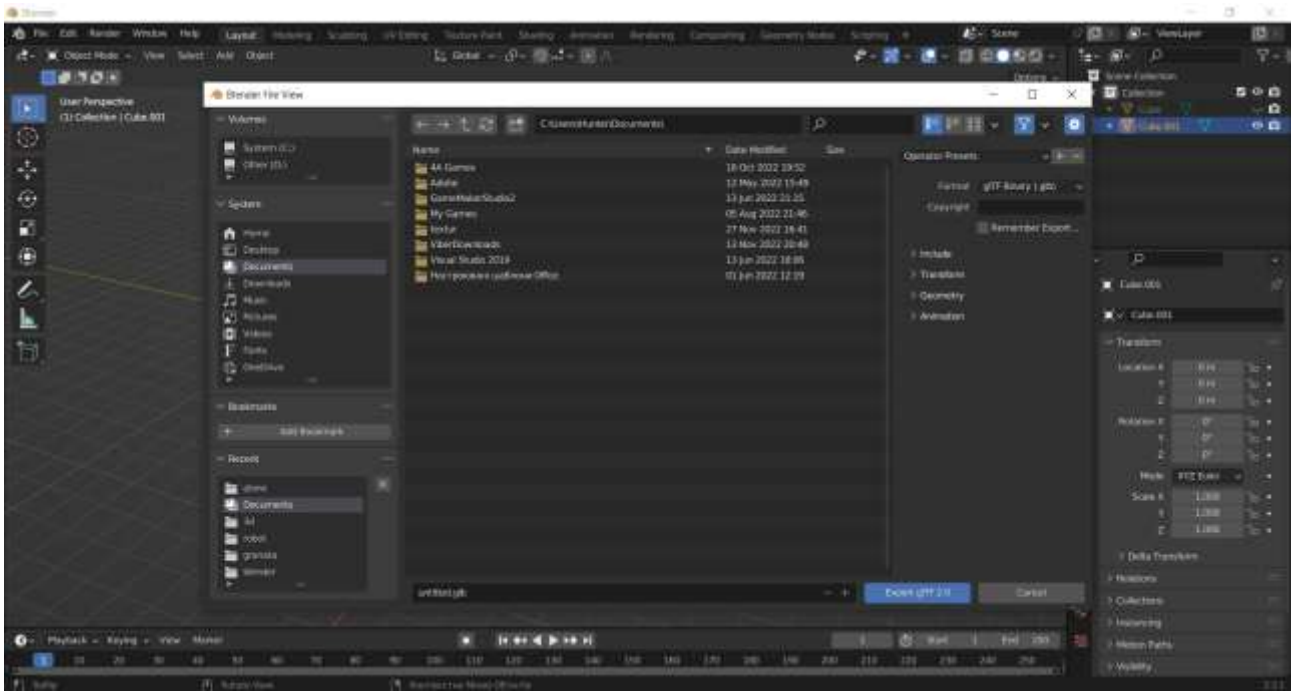


Рис 3.6. Експорт 3D моделі для Godot

Експортування для Unity відбувається за таким самим принципом, тільки інший формат – «FBX». Потрібно натиснути підтвердження у пункті

«Selected Objects» для того, щоб випадково не експортувати ті моделі, які нам не потрібні (рисунок 3.7).

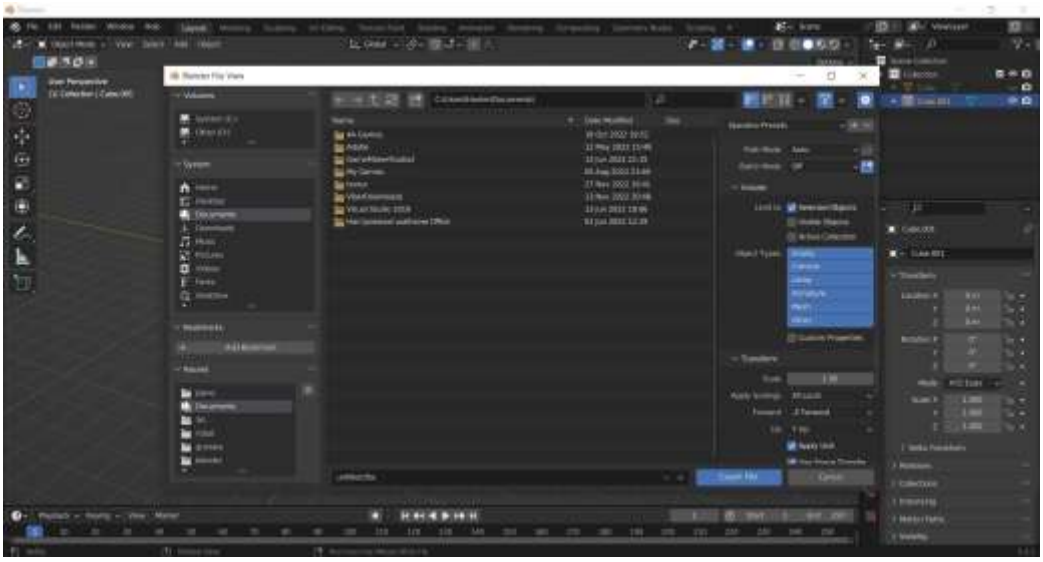


Рис. 3.7. Експорт 3D моделі для Unity

3.2 Розробка ігрового прототипу в ігровому рушії Unity

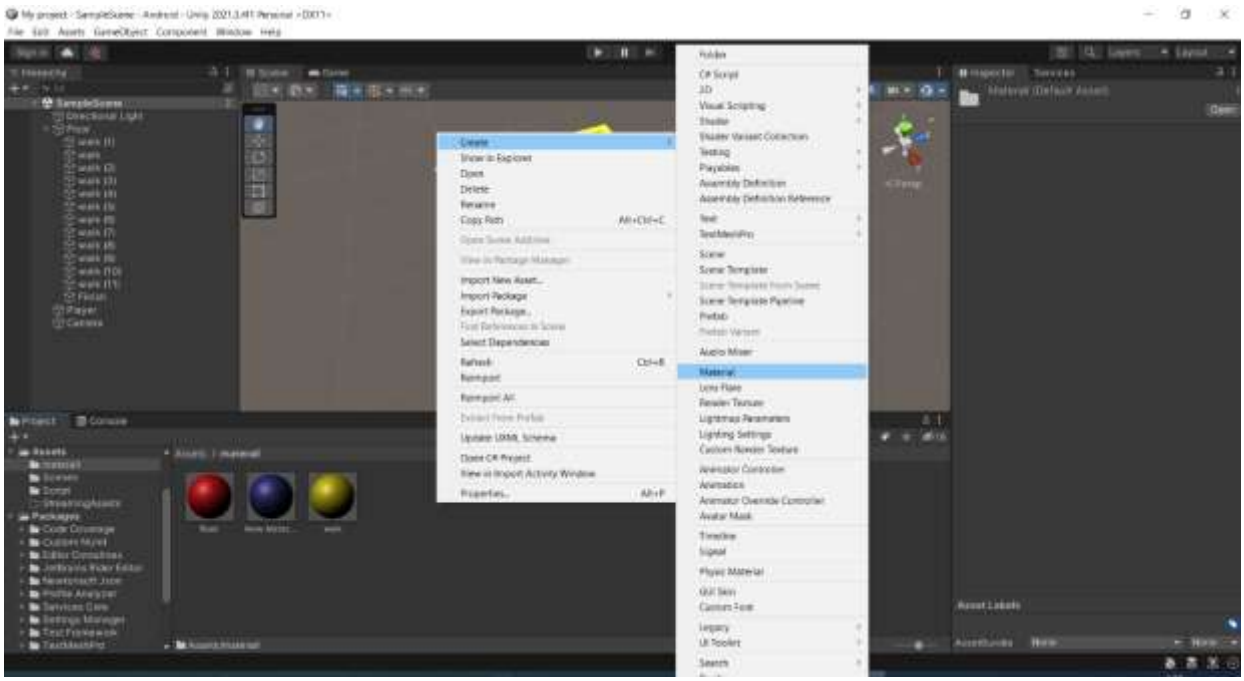
Суть гри: потрібно провести шар через лабіринт до виходу. Модель гравця експортуємо з «Blender». Для зручності створюємо папки, в яких будемо працювати (рисунок 3.8).

Рис. 3.8. Створення папок

Були створені об’єкти які потрібні для гри (рисунок 3.9). Щоб додати їх по правій кнопці мишки обираємо пункт 3d object та після цього обираємо потрібну фігуру.

Рис. 3.9. Об’єкти, які використані в сцені

Після додавання 3D-моделей потрібно створити матеріал для кращого візуального оформлення гри. Потрібно зайти в папку «Матеріали», натиснути праву кнопку мишки, в табличці яка з'явиться натиснути «Create» далі «Material» (рисунок 3.10). Щоб надати колір матеріалу потрібно натиснути на



пункт «Albedo» і обрати колір (рисунок 3.11).

Рис. 3.10. Додавання матеріалу

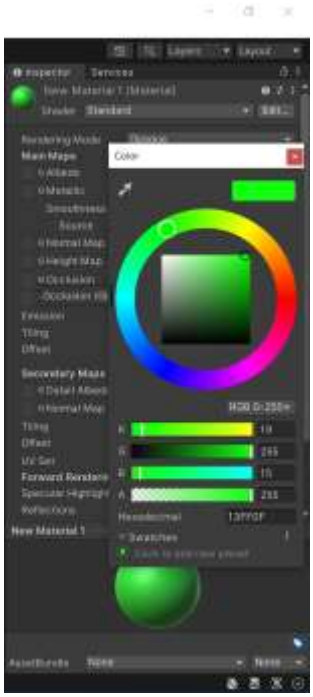
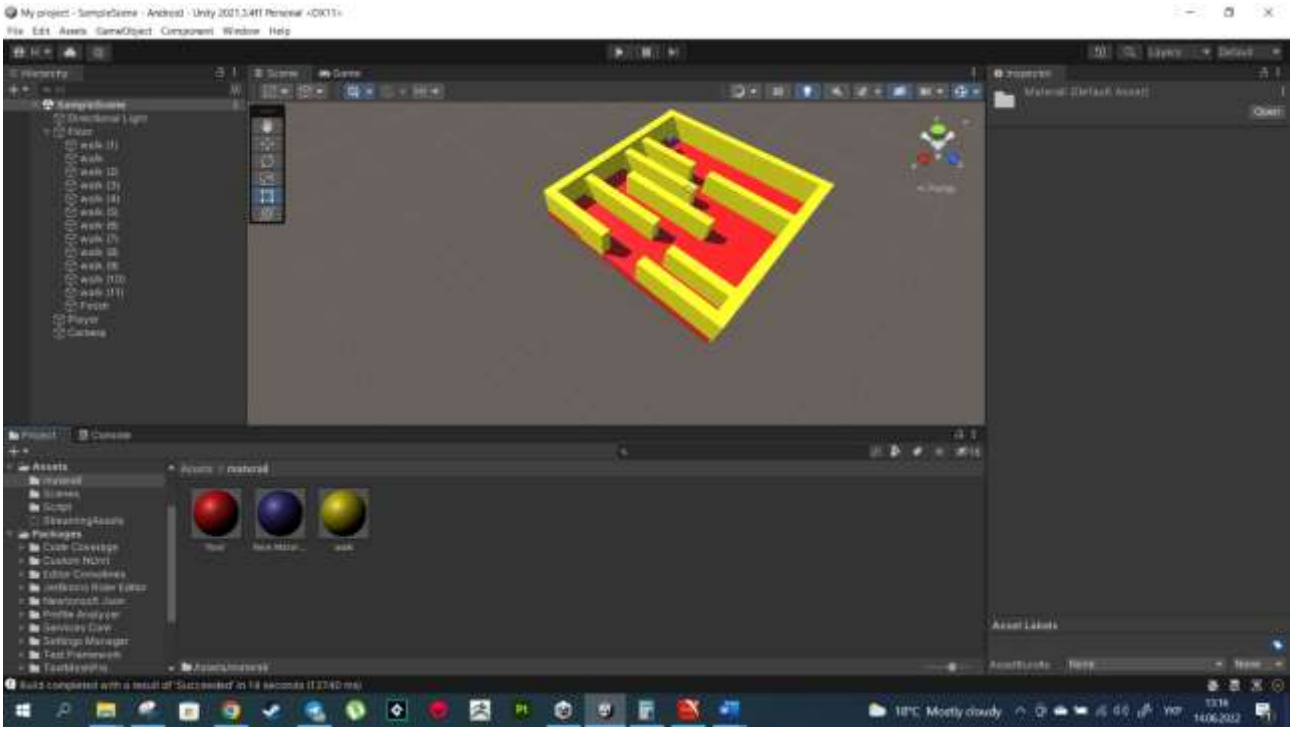


Рис. 3.11. Колір матеріалу

Створивши ігрове середовище, початкову позицію гравця, розробив



лабіринт, також добавив матеріали на об’єкти (рисунок 3.12).

Рис. 3.12. Побудова лабіринту

Зробив скриптом управління для гри на ОС Android (рисунок 3.12). Скрипт – це програма або програмний файл сценарій, які автоматизують деяку задачу, яку користувач робив би вручну, використовуючи інтерфейс програми. Скрипти пишуться на скриптових мовах, які відрізняються за своїм синтаксисом.

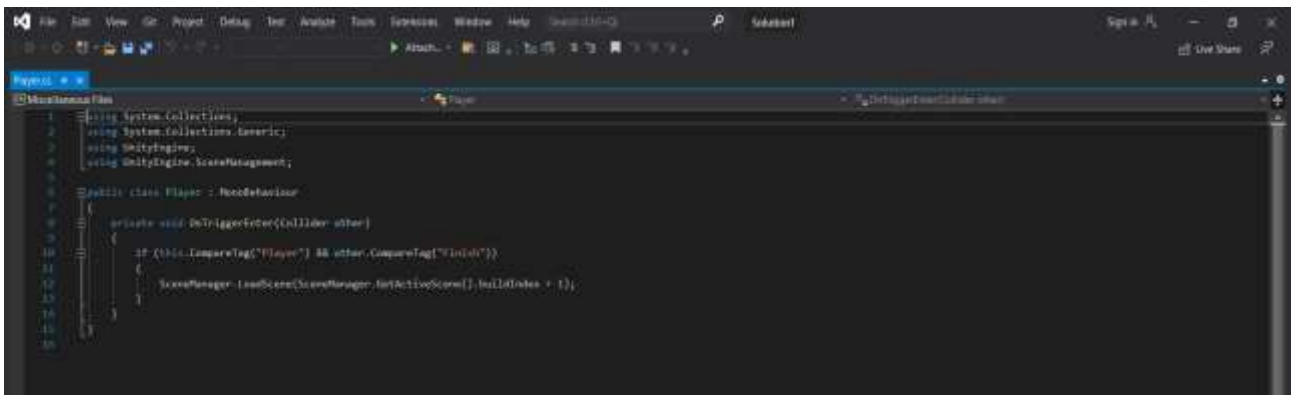


Рис. 3.12. Управління гри

Також прописав фізику і логіку гри (рисунок 3.13).

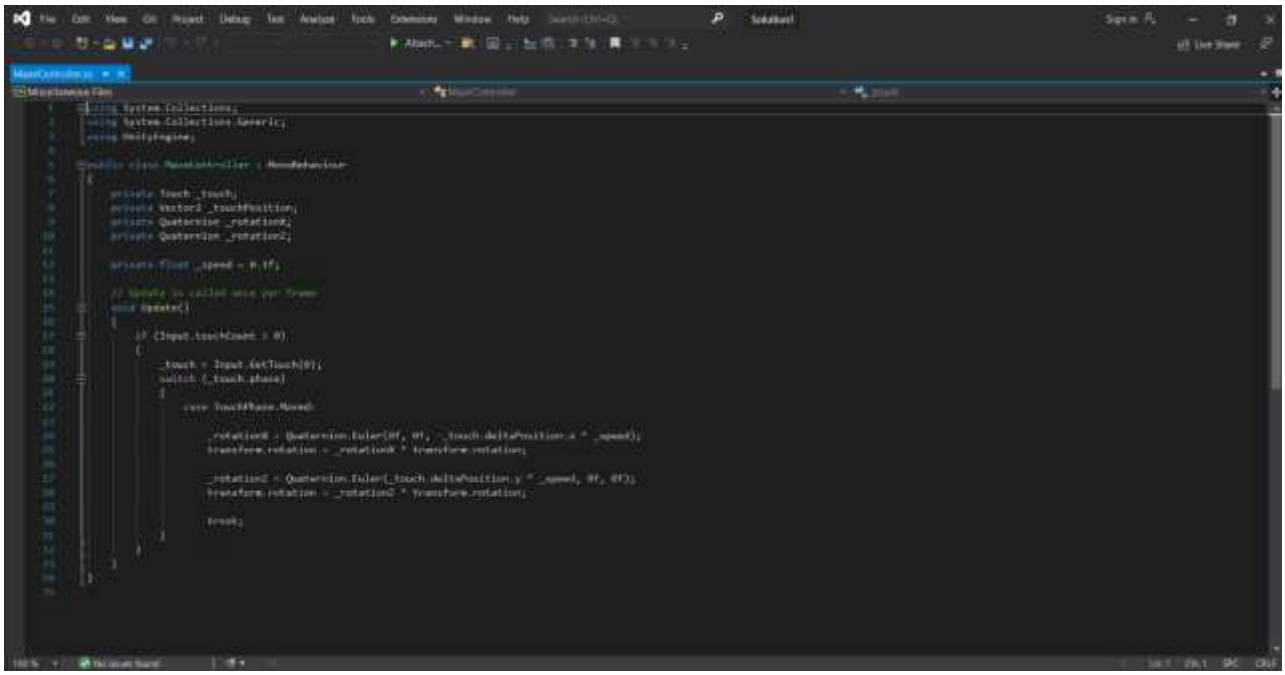


Рис. 3.13. Фізика гри

Здійснив портування проекту на Android. Портування – модифікація програмного забезпечення для перенесення з однієї апаратної платформи на іншу або між різними операційними системами. Результат портування називають портом. Необхідність портування операційної системи виникає при перенесенні її на комп'ютери з іншою архітектурою та іншим типом процесора з іншим набором команд. Портування потрібне також у випадку перенесення певних програмних продуктів, наприклад, відеоігор, з персонального комп'ютера на кишеньковий, мобільний телефон тощо.

Для портування проекту на Android, потрібно встановити Android Studio, після чого в Unity можна імпортувати готовий застосунок (рисунок 3.14).

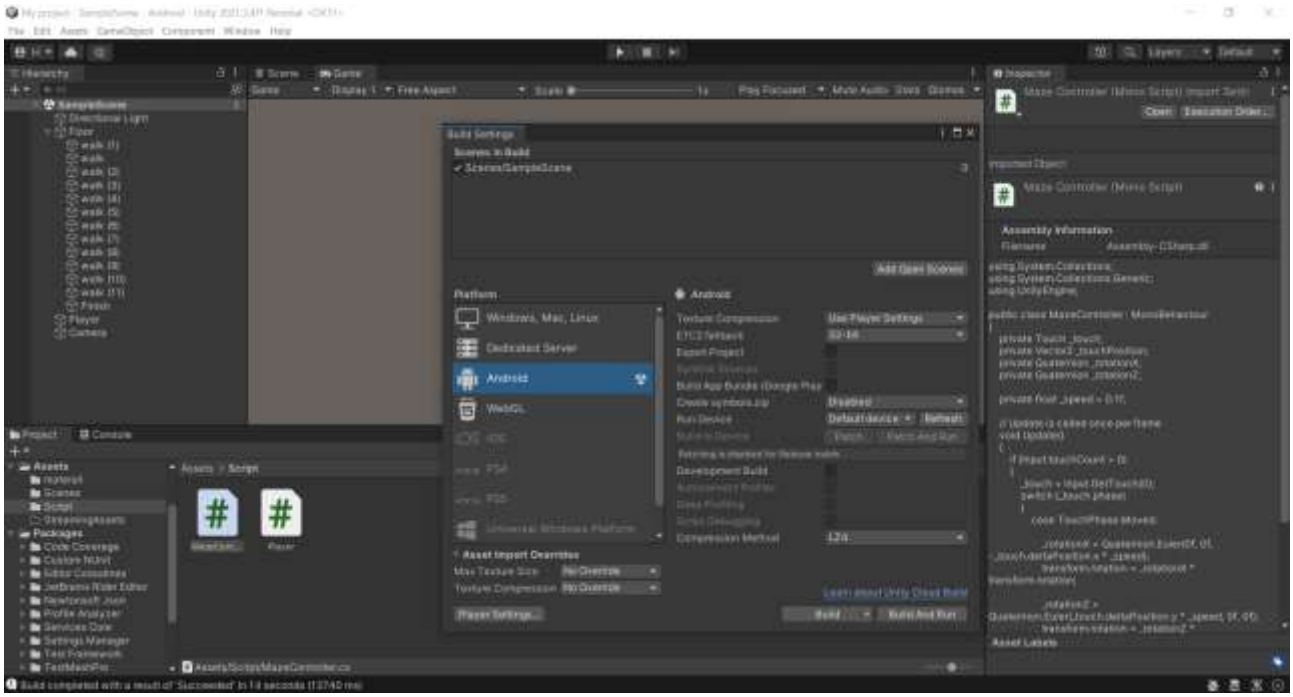


Рис. 3.14. Портування проекту

Створив різні рівні, змінюючи лабіринт (рисунок 3.15).

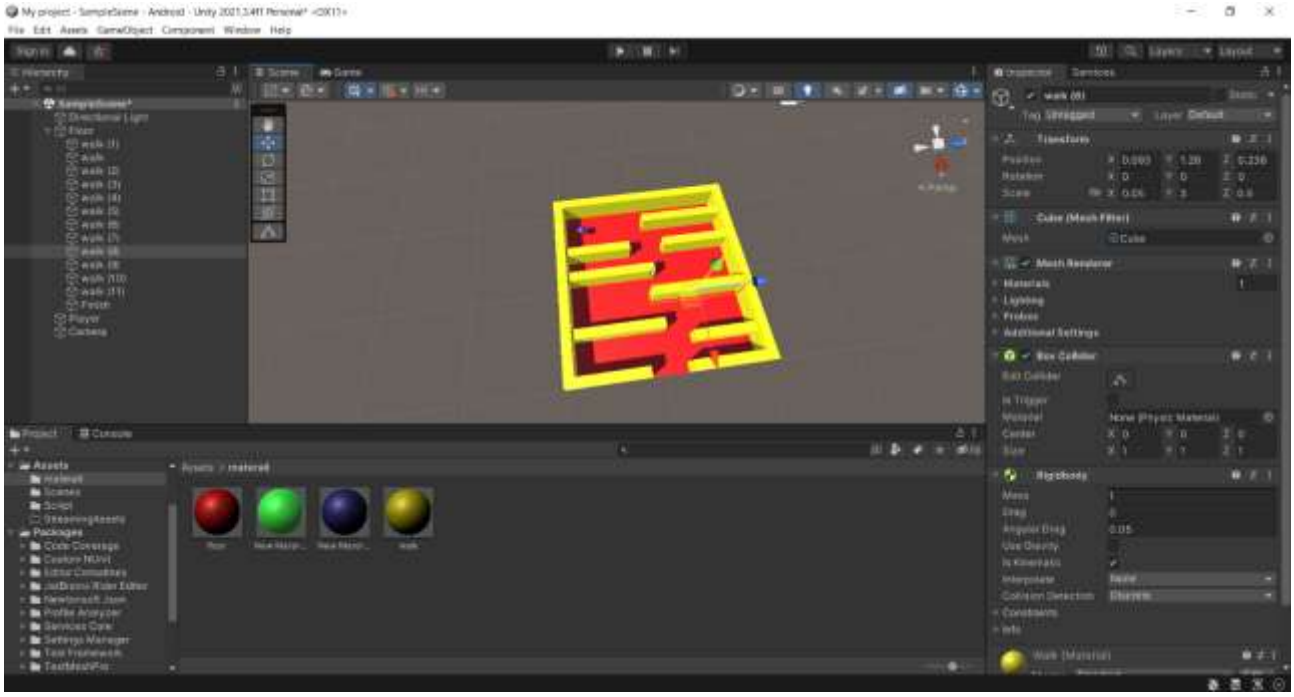


Рис. 3.15. 2-й рівень гри

Фінальний результат показано на рисунку 3.16.

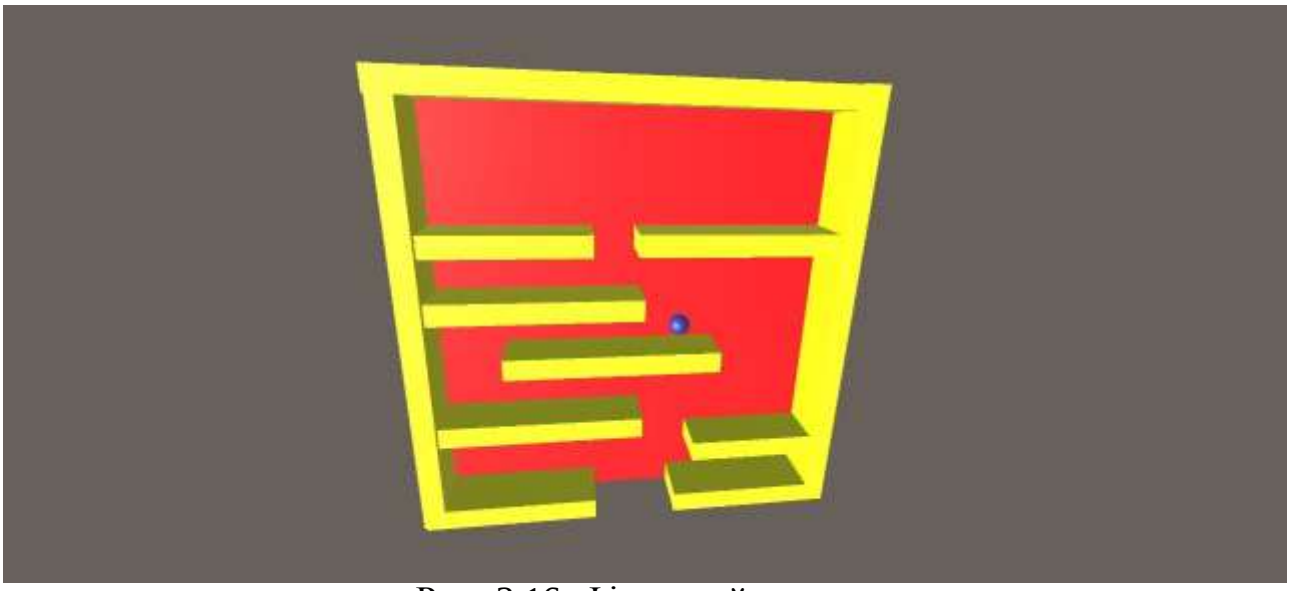


Рис. 3.16. Фінальний результат

3.3. Розробка ігрового прототипу засобами ігрового рушія Godot Engine

Суть гри: потрібно провести шар по дорозі, щоб він не випав. Експортуємо модель гравця з «Blender». Для зручності створюємо папки, в яких будемо працювати (рисунок 3.17).

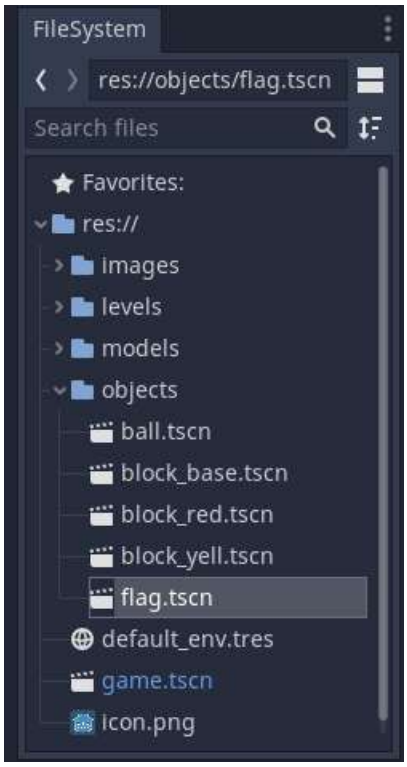


Рис. 3.17. Об’єкти для гри

Створюємо об’єкт «м’яч», налаштовуємо його фізику, а також колізію (рисунки 3.18-3.19).

Рис. 3.18. Налаштування об’єкту гравець

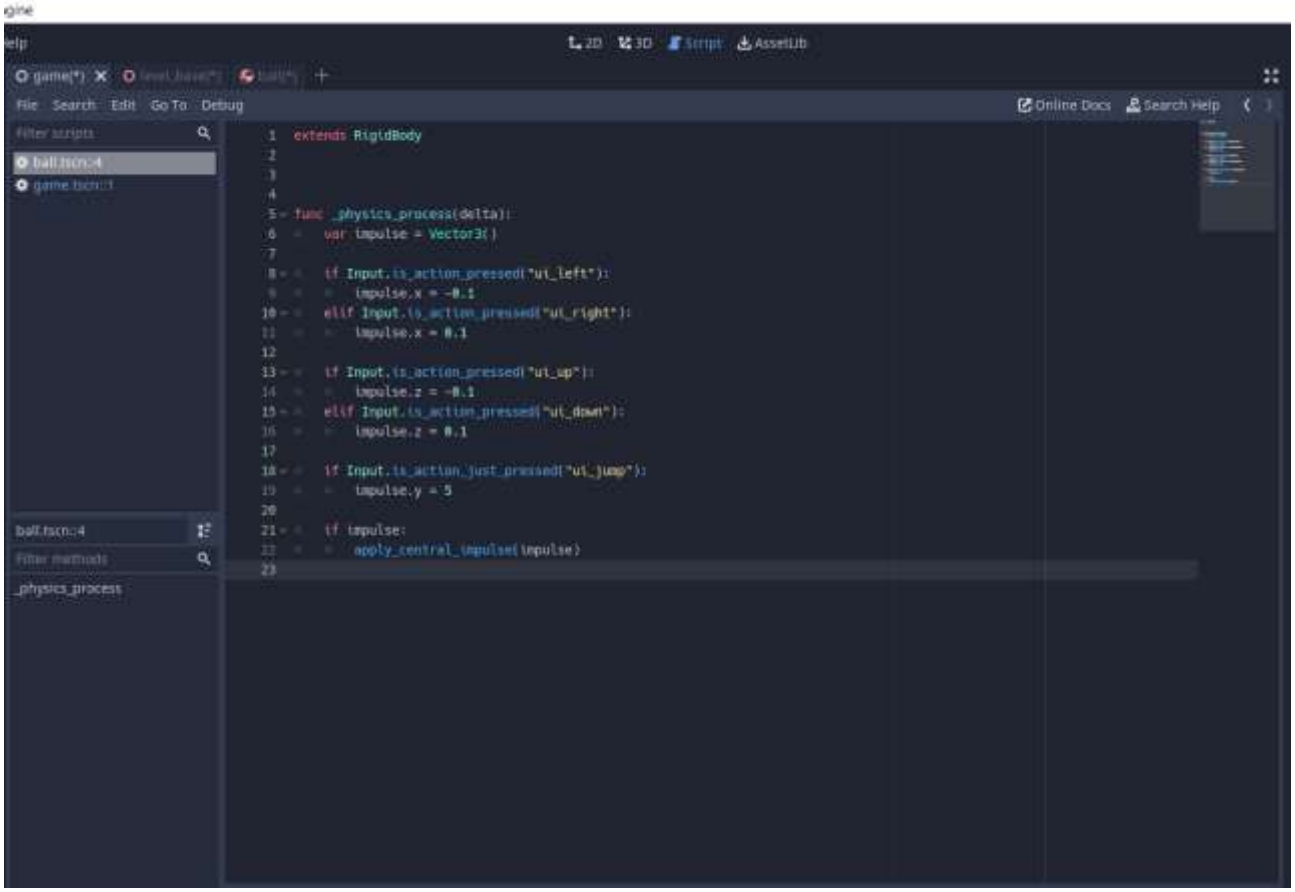


Рис. 3.19. Керування для гравця

Додаємо камеру та пишемо скрипт, щоб вона слідувала за гравцем. Скрипт камери подано на рисунку 3.20.

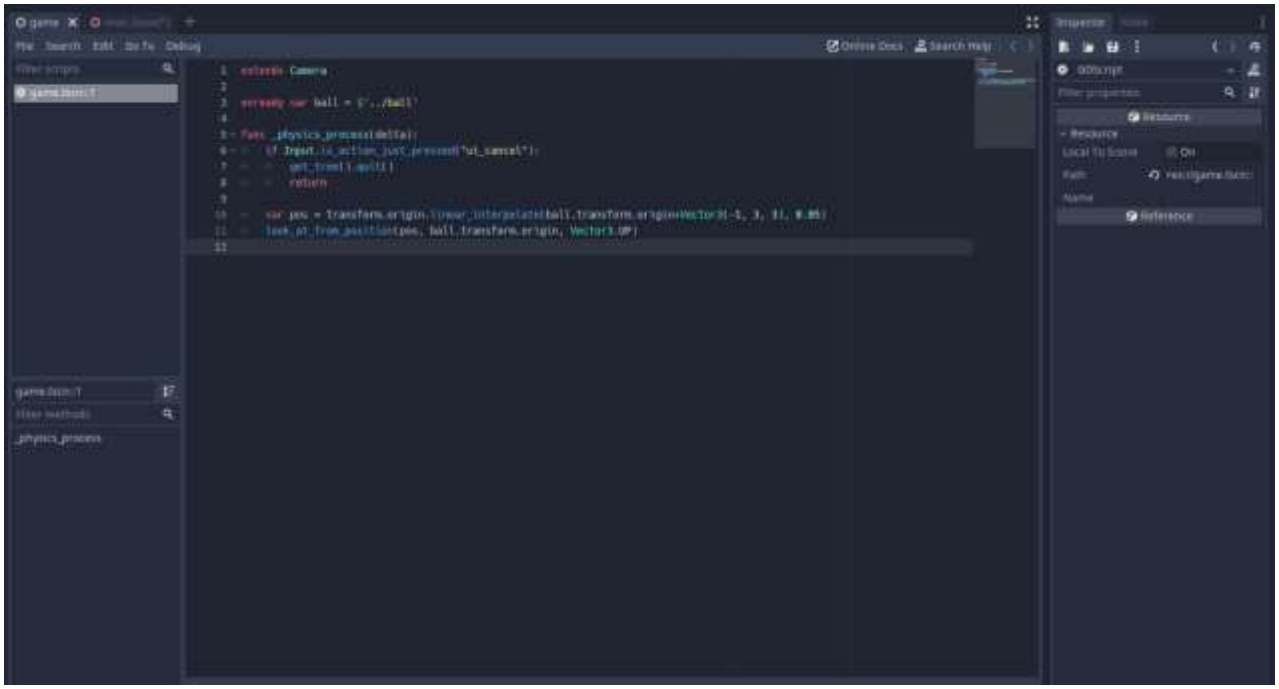


Рис. 3.20. Скрипт камери

Здійснюємо налаштування, щоб при зіткненні з прапором рівень перезавантажувався (рисунок 3.21). Потім можна добавити ще рівнів.

Рис. 3.21. Скрипт взаємодії прапора

Таким чином, перший рівень створено (рисунок 3.22).

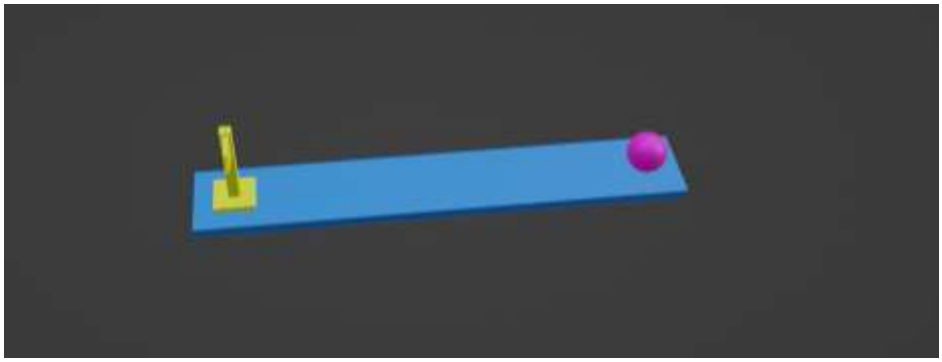


Рис. 3.22. 1-й рівень гри

На рисунку 3.23 показано 2-ий рівень гри.

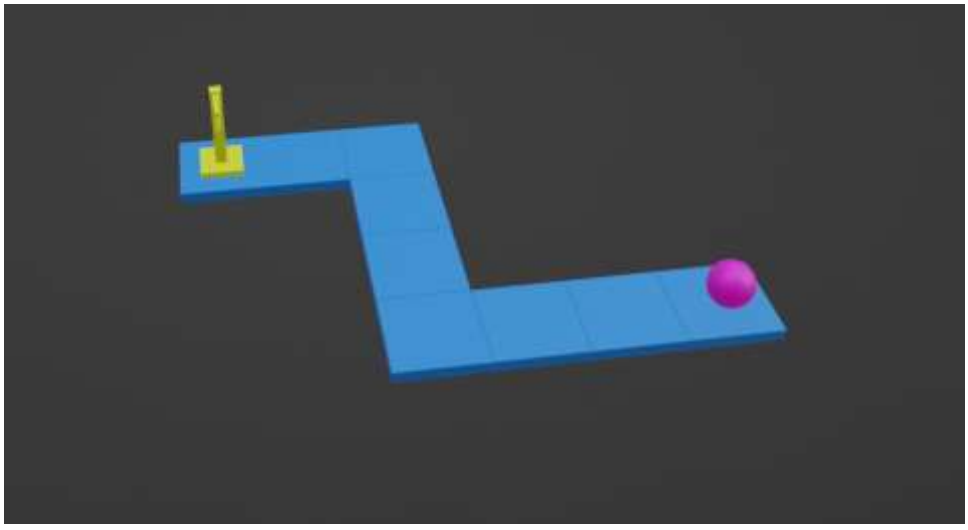


Рис 3.23. 2-й рівень гри

Для експортування проекту на андроїд потрібно в лівому верхньому куті натиснути «Project» далі «Export», після цього «Add» та обрати платформу Android, після чого натиснути клавішу Export/Zip (рисунок 3.24).

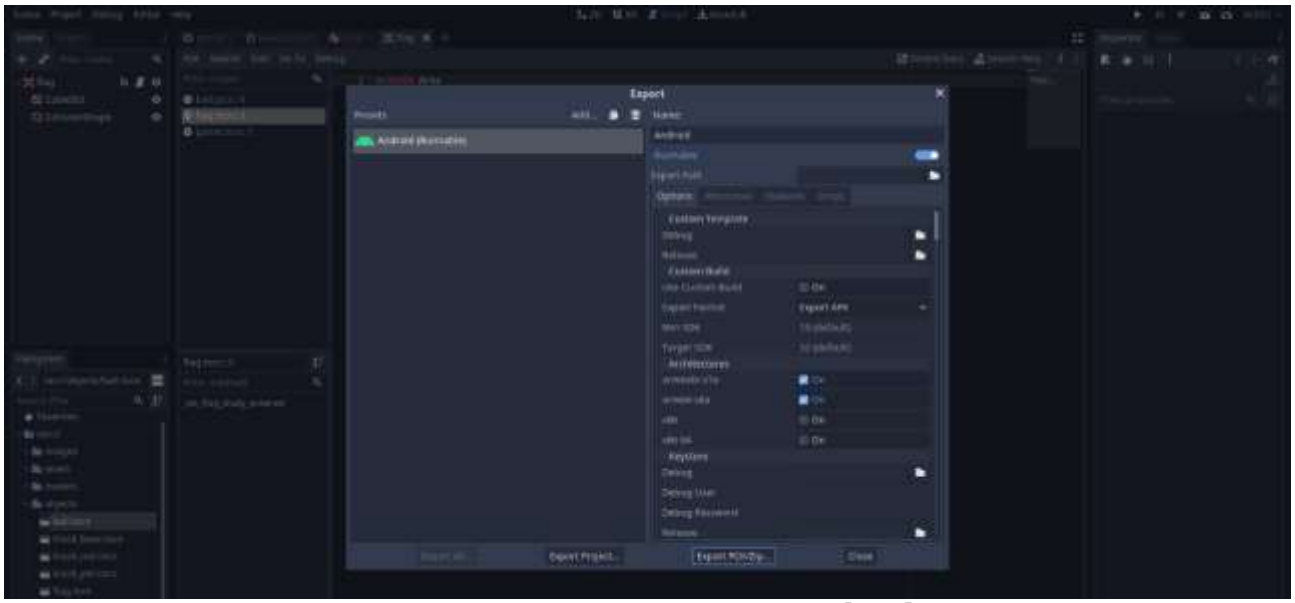


Рис. 3.24. Експорт на Android

Реалізовано лише базову механіку в ігрових прототипах, жодних складних текстур, анімації та звуків. Але цю гру легко можна використовувати як базу під час розробки.

ВИСНОВКИ

Комп’ютерні ігри вже тривалий час є частиною нашого життя. Кількість їх жанрів і різновидів дедалі важче перелічити. На сьогоднішній день ринок комп'ютерних ігор задає досить високу планку для розробників ігрових додатків. Сучасна гра повинна мати сильні сторони в області графіки та цікавого геймплея.

Індустрія розробки ігор стає все популярнішою в сучасному світі та набуває все більшого поширення в нашому суспільстві. Ігри є не просто темою розваги, а й використовуються в наш час в багатьох інших галузях, таких як наука або навчання користувачів. Тому напрямок розвитку ігрової індустрії є популярним і актуальним в сучасному суспільстві.

Під час роботи над дипломним проектом був проведений детальний аналіз предметної області мобільних ігрових додатків, а також було проаналізовано різні джерела інформації щодо розробки програмних продуктів для платформи Android.

Також в роботі проаналізовано жанри мобільних додатків. Проведено дослідження методів та засобів їх реалізації.

Під час ознайомлення з популярними ігровими рушіями та вивчення їх можливостей здійснено дослідження та порівняльну характеристику їх функціональних можливостей, а також визначено переваги та недоліки цих платформ. Для розробки ігрових прототипів було досліджено ігрові рушії Unity та Godot. Можна зробити висновок, що Unity є одним з лідерів в індустрії серед ігрових рушіїв. На даний момент Godot сильно відстає від свого конкурента.

Здійснено протипування ігрового додатка. В результаті розроблено два мобільних ігрових додатки в жанрі аркада. Створено головний ігровий цикл, 3D-моделі, інтерфейс додатку та візуальні ефекти.

Розглянуті платформи мають низький поріг входження, тобто не передбачають наявності масштабної бази знань у розробника. Вони мають

зручний, інтуїтивно зрозумілий інтерфейс, що сприяє швидкому вивченню різноманітності елементів ігрових рушіїв. Також, постійна підтримка розробників та своєчасне оновлення функціоналу дає змогу створити у найкоротший термін стабільний, якісний продукт, який відповідає сучасним запитам.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Джо Хокінг Unity в дії. Мультиплатформенна розробка на C # [Текст].
2. Godot docs: Coroutines [Електронний ресурс] https://docs.godotengine.org/en/stable/index.html
3. Unity - Manual: Coroutines [Електронний ресурс] – https://docs.unity3d.com/Manual/UsingTheEditor.html
4. Експорт FBX [Електронний ресурс] – https://docs.unity3d.com/ru/current/Manual/HOWTO-exportFBX.html
5. Blender 3.3 Reference Manual – https://docs.blender.org/manual/en/latest/
6. Blender [Електронний ресурс] – https://www.blender.org /
7. Infographic: Mobile Game Market Trends 2020 [Електронний ресурс] // Dot Com Infoway. – 2020. – Режим доступу до ресурсу: https://www.dotcominfoway.com/blog/infographic-mobile-game-market-trends-2020/#gref
8. Video Games Could Be a $300 Billion Industry by 2025 (Report) [Електронний ресурс] // Vervelogic. – 2019. – Режим доступу до ресурсу: https://variety.com/2019/gaming/news/video-games-300-billion-industry-2025-report-1203202672/
9. Філософія архітектури Godot [Режим доступу: https://docs.godotengine.org/uk/stable/getting_started/introduction/godot_design_philosophy.html]
10. About Blender [Режим доступу: https://docs.blender.org/manual/uk/2.82/getting_started/about/introduction.htm]