

**МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ**

**ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І
СИСТЕМ**

КВАЛІФІКАЦІЙНА РОБОТА

**на тему: “ПРОЄКТУВАННЯ ТА РОЗРОБКА ГРИ У ЖАНРІ СТРАТЕГІЯ
В СЕРЕДОВИЩІ GAMESMAKER”**

Здобувача вищої освіти першого (бакалаврського) рівня,
IV курсу, групи ЦДБ-22-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Технології цифрового дизайну»



(підпис)

Пригоровського Владислава
Денисовича

(прізвище, ім'я, по батькові)

Науковий керівник



(підпис)

Гладченко Оксана Вікторівна,
канд. пед. наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Керівник
освітньо-професійної
програми



(підпис)

Ратушняк Тетяна Володимирівна,
канд. фіз.-мат. наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Допущено до захисту за рішенням кафедри комп'ютерних та інформаційних
технологій і систем протокол від «12» травня 2026 №20

Завідувач кафедри



(підпис)

Філоненко Михайло
Миколайович, канд. фіз.-мат.
наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Ірпінь – 2026

Реєстрація № 73

29.05.2026
(дата)


(підпис відповідальної особи)

Результати перевірки

До захисту
(до захисту, на доопрацювання)

12.05.2026 
(дата) (підпис)

Результат захисту

5 (відмінно)
(оцінка)

11.06.2026 
(дата) (підпис)

Голова комісії:


(підпис)

О.С. Дюбанов

Члени комісії:


(підпис)

М.М. Філоненко


(підпис)

Т.В. Ратушняк


(підпис)

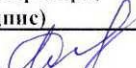
В.О. Ніжегородцев

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії, секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	29		11.06.2026 Гладченко О.В.  Дюбанов О.С.  Філоненко М.М.  Ратушняк Т.В.  Ніжегородцев В.О.  Суховерхня Л.П. 
2. Дотримання вимог щодо оформлення	від 0 до 10	9		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	4		
4. Рецензент	від 0 до 5	5		
РАЗОМ БАЛІВ	від 0 до 50	47		

Оцінювання кваліфікаційної роботи науковим керівником – 47 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підписи)
94	A	5 (відмінно)	11.06.2026	Дюбанов О.С.  Філоненко М.М.  Ратушняк Т.В.  Ніжегородцев В.О.  Суховерхня Л.П. 

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	
50-59	E	3 (задовільно)
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань: 12 «Інформаційні технології»
Спеціальність: 122 «Комп'ютерні науки»
Освітня програма: «Технології цифрового дизайну»

Затверджую
Завідувач кафедри  Філоненко М.М.
«01» грудня 2025

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Пригоровському Владиславу Денисовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи Проектування та розробка гри у жанрі стратегія в середовищі GameMaker

Затверджена наказом ректора від «28» листопада 2025 № 1526

2. Строк здачі закінченої роботи «11» травня 2026

3. Мета роботи полягає у розробці та програмній реалізації відеогри у жанрі економічна стратегія типу «Монополія» в середовищі GameMaker із повною автоматизацією ігрових механік та фінансових розрахунків.

Об'єкт дослідження Процес проектування та програмного моделювання ігрових систем у цифровому середовищі.

Предмет дослідження Програмні алгоритми автоматизації ігрової логіки та інструментарій розробки в середовищі GameMaker.

Орієнтовна база дослідження Інформаційне середовище та технічні потужності рушія розробки GameMaker Studio.

4. Вихідні дані до роботи Технічна документація середовища розробки GameMaker, стандарти проектування інтерфейсів користувача, правила та логічна структура класичної настільної гри «Монополія».

5. Перелік графічного матеріалу Блок-схема алгоритму гри, зображення елементів внутрішньоігрового інтерфейсу, скріншоти середовища рушія GameMaker.

6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)

РОЗДІЛ 1. ТЕОРЕТИЧНІ ВІДОМОСТІ ТА ПІДГОТОВКА. 1.1. Дослідження літератури та існуючих ігрових аналогів. 1.2. Аналіз можливостей середовища GameMaker. 1.3. Проектування ігрової концепції та підготовка ресурсів. РОЗДІЛ 2. ПРАКТИЧНИЙ ЕТАП СТВОРЕННЯ ПРОГРАМНОГО ПРОДУКТУ. 2.1. Розробка графічних ресурсів та архітектури об'єктів у середовищі GameMaker. 2.2. Реалізація ключових ігрових механік та логіки взаємодії. 2.3. Інтеграція ШІ, звуку та фінальні доопрацювання. РОЗДІЛ 3. АНАЛІЗ РЕЗУЛЬТАТУ РОЗРОБКИ, ПУБЛІКАЦІЯ ТА ПЕРСПЕКТИВИ РОЗВИТКУ. 3.1. Аналіз та оцінка результату роботи. 3.2. Підготовка та формування фінального програмного пакета. 3.3. Аналіз можливостей вдосконалення та перспектив розвитку гри. ВИСНОВОК. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	19.12.2025	19.12.2025
2	I розділ (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	25.02.2026	28.02.2026
3	II розділ (аналіз експериментальної частини про методики досліджень)	12.03.2026	29.03.2026
4	III розділ (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	01.04.2026	10.04.2026
5	Висновки	16.04.2026	16.04.2026
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	24.04.2026	26.04.2026
7	Отримання відгуку керівника	01.05.2026	01.05.2026
8	Отримання зовнішньої рецензії	04.06.2026	04.06.2026

8. Дата видачі завдання «01» грудня 2025

9. Керівник кваліфікаційної роботи


(підпис)

Гладченко О.В.
(прізвище, ініціали)

10. Керівник освітньо-професійної програми


(підпис)

Ратушняк Т.В.
(прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти


(підпис)

Пригоровський В.Д.
(прізвище, ініціали)

АНОТАЦІЯ

Проектування та розробка гри у жанрі стратегія в середовищі GameMaker

(назва кваліфікаційної роботи)

Мета роботи – розробка відеогри у жанрі економічна стратегія типу «Монополія» в середовищі рушія GameMaker. Завданнями є аналіз існуючих аналогів, проектування архітектури ігрових об'єктів та логіки взаємодії, реалізація алгоритмів автоматизованого переміщення, індексації клітинок поля і нарахування оренди, а також тестування системи. Об'єктом дослідження є процес проектування та програмного моделювання ігрових систем у цифровому середовищі. Предмет дослідження це програмні алгоритми автоматизації ігрової логіки та архітектурні рішення середовища GameMaker. Базою дослідження виступає інформаційне середовище розробки GameMaker Studio та результати аналізу економічних стратегій.

За результатами роботи побудовано ієрархію ігрових об'єктів, реалізовано алгоритми автоматизації фінансових розрахунків та модель інтерактивного інтерфейсу користувача. Тестування підтвердило стабільність виконання транзакцій та точність логіки ходів ігрових персонажів. Отримані результати можуть застосовуватися як основа для створення подібних настільних ігор у цифровому просторі або як навчальний симулятор для вивчення основ програмування на GML та теорії економічних стратегій.

Кваліфікаційна робота містить 50 сторінок, 0 таблиць, 17 рисунків, список використаних джерел із 40 найменувань, 0 додатків.

Ключові слова: GAMEMAKER, GML, ЕКОНОМІЧНА СТРАТЕГІЯ, МОНОПОЛІЯ, ІГРОВА ЛОГІКА, АВТОМАТИЗАЦІЯ, ІЄРАРХІЯ ОБ'ЄКТІВ, ІНТЕРФЕЙС КОРИСТУВАЧА

Рік виконання кваліфікаційної роботи: 2026.

Рік захисту роботи: 2026.

ABSTRACT

Design and development of a strategy genre game in the GameMaker environment

(title of the qualification work)

The objective of the research is the development and software implementation of a video game in the economic strategy genre similar to "Monopoly" within the GameMaker engine environment. The tasks include the analysis of existing analogues, design of the game object architecture and interaction logic, implementation of algorithms for automated movement, field cell indexing, and rent calculation, as well as system testing. The object of the study is the process of designing and software modeling of game systems in a digital environment. The subject of the study includes software algorithms for game logic automation and architectural solutions of the GameMaker environment. The research framework is based on the GameMaker Studio development environment and the results of economic strategy analysis.

Based on the research results, a hierarchy of game objects was constructed, financial calculation automation algorithms were implemented, and an interactive user interface model was developed. Testing confirmed the stability of transaction execution and the accuracy of game character movement logic. The obtained results can be applied as a foundation for creating similar board games in digital space or as an educational simulator for studying the basics of GML programming and the theory of economic strategies.

The qualification work consists of 50 pages, 0 tables, 17 figures, a list of cited sources with 40 items, 0 appendices.

Keywords: GAMEMAKER, GML, ECONOMIC STRATEGY, MONOPOLY, GAME LOGIC, AUTOMATION, OBJECT HIERARCHY, USER INTERFACE.

Year of qualification work completion: 2026.

Year of work defense: 2026.

ЗМІСТ

ВСТУП.....	3
РОЗДІЛ 1. ТЕОРЕТИЧНІ ВІДОМОСТІ ТА ПІДГОТОВКА.....	5
1.1. Дослідження літератури та існуючих ігрових аналогів	5
1.2. Аналіз можливостей середовища GameMaker	6
1.3. Проєктування ігрової концепції та підготовка ресурсів	9
РОЗДІЛ 2. ПРАКТИЧНИЙ ЕТАП СТВОРЕННЯ ПРОГРАМНОГО ПРОДУКТУ	16
2.1. Розробка графічних ресурсів та архітектури об'єктів у середовищі GameMaker	Помилка! Не вказано ім'я закладки.
2.2. Реалізація ключових ігрових механік та логіки взаємодії	22
2.3. Інтеграція ШІ, звуку та фінальні доопрацювання	37
РОЗДІЛ 3. АНАЛІЗ РЕЗУЛЬТАТУ РОЗРОБКИ, ПУБЛІКАЦІЯ ТА ПЕРСПЕКТИВИ РОЗВИТКУ	40
3.1. Аналіз та оцінка результату роботи	Помилка! Не вказано ім'я закладки.
3.2. Підготовка та формування фінального програмного пакета.....	41
3.3. Аналіз можливостей вдосконалення та перспектив розвитку гри.....	42
ВИСНОВОК.....	45
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	47

ВСТУП

Актуальність теми. Стрімка цифровізація суспільства та інтеграція програмних рішень у сферу розваг зробили перехід класичних настільних ігор у цифровий простір необхідним етапом їх розвитку. Створення відеогри типу «Монополія» дозволяє повністю автоматизувати складні фінансові розрахунки, уникнути помилок при обробці правил та забезпечити динамічну взаємодію гравців незалежно від їхнього місця розташування. З технічної точки зору, розробка такого проєкту є важливою для дослідження методів проєктування ігрових систем у середовищах швидкої розробки, що поєднують складну математичну логіку та інтерактивний інтерфейс.

Стан і ступінь розробки проблеми. Ігрова індустрія сьогодні є одним із найбільш інноваційних секторів ІТ, де активно досліджуються питання об'єктно-орієнтованого моделювання та ігрового дизайну. Хоча на ринку існують масштабні цифрові версії «Монополії», питання створення гнучкої та легкої архітектури для незалежних розробників у середовищі GameMaker залишається актуальним.

Метою кваліфікаційної роботи є розробка та програмна реалізація відеогри «Монополія» за допомогою середовища GameMaker, що забезпечує автоматизацію ігрових подій, прозорість фінансових транзакцій та зручну візуальну взаємодію користувачів.

Для досягнення мети визначено наступні завдання: вивчити технічний інструментарій середовища GameMaker та можливості мови GML, спроектувати програмну архітектуру проєкту на основі ієрархії об'єктів, реалізувати алгоритми автоматизованого переміщення та індексації клітинок поля, розробити економічну модель гри з функціями нарахування оренди та управління активами, а також обґрунтувати структуру інтерфейсу користувача для відображення ігрових подій у реальному часі.

Об'єктом дослідження є процес проєктування, технічного моделювання та програмної реалізації ігрових систем у цифровому середовищі. Це включає вивчення загальних принципів побудови ігрової логіки, розробку механік взаємодії між

об'єктами та впровадження систем, що забезпечують стабільне функціонування економічних стратегій у межах спеціалізованих ігрових рушіїв.

Предметом дослідження є програмні алгоритми автоматизації ігрової логіки, функціональні можливості мови програмування GML та специфічні архітектурні рішення середовища GameMaker. Основна увага приділяється методам побудови ієрархії об'єктів, реалізації математичних моделей для фінансових розрахунків та створенню механізмів обробки випадкових подій, які складають технічну основу розробленого проєкту.

Методи дослідження. У роботі використано системний аналіз для вивчення ігрових механік, методи об'єктно-орієнтованого програмування для побудови структури гри, а також метод тестування для перевірки коректності виконання фінансових транзакцій та логіки ходів.

База досліджень це інформаційне середовище та технічні потужності середовища розробки GameMaker Studio, відкриті бібліотеки графічних ресурсів, а також результати аналізу існуючих настільних та цифрових економічних стратегій.

Наукова та практична значимість одержаних результатів. Створений ігровий додаток має високий ступінь автоматизації, що дозволяє використовувати його як базу для подальших комерційних розробок або як навчальний симулятор для вивчення основ економічної стратегії. Практична цінність полягає у розробленій архітектурі взаємодії об'єктів, яка може бути адаптована для створення інших настільних ігор у цифровому форматі, мінімізуючи складність розрахунків для кінцевого користувача.

Апробація результатів роботи: Основні результати дослідження було апробовано на I Міжнародній науково-практичній конференції здобувачів вищої освіти та молодих вчених «Економіка і бізнес: нові реалії, технологічні тренди, [цифрові інновації]», де вони отримали позитивне обговорення та були враховані у подальшій науковій роботі [1].

РОЗДІЛ 1. ТЕОРЕТИЧНІ ВІДОМОСТІ ТА ПІДГОТОВКА

1.1. Дослідження літератури та існуючих ігрових аналогів

Починаючи роботу над будь-яким складним програмним продуктом, варто приділити особливу увагу етапу попереднього дослідження та аналізу теоретичних праць. Це життєво важливий фундамент, на якому закладається вся подальша розробка проєкту. Створення гри є заплутаним та багатогранним процесом, що об'єднує не лише написання програмного коду, але й математичні обчислення та психологічне розуміння поведінки гравця. Без детального огляду профільної літератури розробник ризикує втратити надмірний час на з'ясування питань, які вже давно були успішно вирішені в індустрії. Саме тому вивчення накопиченого досвіду дає змогу суттєво полегшити процес втілення ігрових механік та зробити логіку проєкту більш досконалою та доступною для кінцевого споживача.

Окремим важливим напрямком дослідження є вивчення основ геймдизайну та поняття ігрового циклу, який часто називають «Core Loop». Для проєкту у жанрі економічної стратегії, такого як «Монополія», цей цикл складається з чітко визначених послідовних дій гравця: кидання кубиків, переміщення фішки по полю, взаємодії з об'єктами нерухомості та подальшого управління фінансовими ресурсами. Спеціалізована література допомагає розробнику зрозуміти, як правильно побудувати ці правила таким чином, щоб гра залишалася цікавою і непередбачуваною протягом тривалого часу та постійно мотивувала користувача до нових дій. Чітке формулювання ігрових механік ще до початку технічної частини розробки дозволяє значно спростити структурування об'єктів та написання сценаріїв у середовищі GameMaker. Такий підхід гарантує, що всі елементи гри будуть працювати як єдиний злагоджений механізм, а розробник зможе уникнути плутанини в коді під час реалізації складних логічних умов та економічних розрахунків.

Великим пластом теоретичної підготовки є опрацювання математичних моделей, які лежать в основі економічних стратегій. Оскільки механіка нашої гри базується на класичній «Монополії», де ключовим складником є випадковість у вигляді кидка гральних кубиків, потрібно чітко розуміти ймовірність випадання тих чи інших комбінацій чисел [40]. Теоретичні джерела з прикладної математики

допомагають розробнику коректно розрахувати вартість ділянок на ігровому полі, розміри орендної плати та податків таким чином, щоб гра не завершувалася занадто швидко і водночас не тривала нескінченно. Вдало налаштований ігровий баланс гарантує, що стратегічні рішення гравця щодо придбання нерухомості матимуть реальне значення, а не залежатимуть виключно від удачі. Це дозволяє зробити ігровий процес у нашому проєкті більш передбачуваним та чесним, що є критично важливим для підтримки зацікавлення користувача до гри.

У процесі теоретичного дослідження особливу увагу було надано опрацюванню фундаментальних праць у сфері геймдизайну. Ці матеріали дають змогу розробнику глибше осягнути структуру ігрових систем та методи утримання інтересу споживачів через коректно встановлені завдань. Окрім друкованих видань, було здійснено докладний огляд популярних електронних аналогів, як наприклад «Monopoly Plus», «Business Tour» та «Rento». Розгляд «Monopoly Plus» дав змогу визначити складні візуальні концепції, тоді як «Business Tour» слугував зразком вдалої імплементації стрімких ігрових партій онлайн. Таке поєднання теоретичних знань із практичним досвідом індустрії є необхідною умовою для створення якісного та конкурентоспроможного проєкту в середовищі GameMaker.

Завершуючи детальний огляд теоретичних джерел та існуючих ігрових рішень, важливо підкреслити, що отримані знання дозволяють сформулювати чітке бачення мого майбутнього проєкту. Вивчення фундаментальних праць та аналіз популярних ігор дали змогу виділити найбільш значущі елементи, які впливають на успіх економічної стратегії.

1.2. Аналіз можливостей середовища GameMaker

Першим етапом початку розробки будь-якої гри є вибір рушія на якій її буде реалізовано, правильний вибір ігрового рушія є дуже важливим для успішного створення гри. Від можливостей обраного рушія залежить швидкість та зручність розробки, коректне функціонування майбутньої гри та можливість її запуску на різних пристроях чи у різних браузерах. Якщо для роботи буде обрано невдалий рушій, це призведе до значних проблем чи обмежень які обов'язково стануть на заваді в процесі створення гри. Саме тому при виборі середовища розробки потрібно не

тільки розібратись, які параметри та які можливості рушія є ключовими для реалізації мого проєкту, а й проаналізувати всі доступні альтернативи та обрати той рушій, який найкраще підходить для розробки гри типу “Монополія”.

Перший параметр ігрового рушія який суттєво вплине на якість кінцевого продукту – це можливості роботи з двовимірною графікою. Хоча настільна гра в жанрі стратегії не потребує складної тривимірної картинки, але вимагає чіткого відображення внутрішнього ігрових подій, фішок, повідомлень тощо. Також дуже важливим є наявність інструментарію для створення зручного інтерфейсу користувача (UI), оскільки ігровий процес Монополії тісно пов’язаний з натисканням кнопок, взаємодією з текстовими чи інтерактивними повідомленнями та іншими елементами гри. Ще потрібно подумати про кросплатформність та кросбраузерність, це означає що гра повинна запускатись та працювати коректно без візуальних викривлень на різних пристроях і браузерах, можливість реалізації цього залежить в тому числі і від ігрового рушія. Наостанок ігровий рушій має бути легким у використанні, інтуїтивно зрозумілим та мати достатній обсяг технічної документації яка допоможе розв’язати ту чи іншу проблему яка може виникнути на етапі розробки. Очевидно, що жоден ігровий рушій не може мати одразу всі переваги та жодних недоліків, тому наступним кроком є аналіз існуючих рушіїв, зокрема GameMaker, порівняння їх між собою та вибір оптимального з них.

Середовище розробки GameMaker одразу привертає особливу увагу, оскільки його функціонал дуже добре підходить для створення саме двовимірних ігор, в тому числі й стратегій типу Монополії. GameMaker має власну мову програмування GML (GameMaker Language) яка є досить простою у використанні навіть для розробників новачків. Також значною перевагою цього рушія є зручний та інтуїтивно зрозумілий інтерфейс, що є важливим для успішної та ефективної розробки продукту. GameMaker, хоча є досить простим рушієм, має весь необхідний функціонал для гнучкого налаштування ігрового процесу і логіки взаємодії між гравцями та подіями на ігровій дошці. Враховуючи ці дані, середовище GameMaker здається оптимальним вибором, оскільки його переваги у розробці 2D-гри є очевидними, а недоліки – навпаки, не настільки суттєвими.

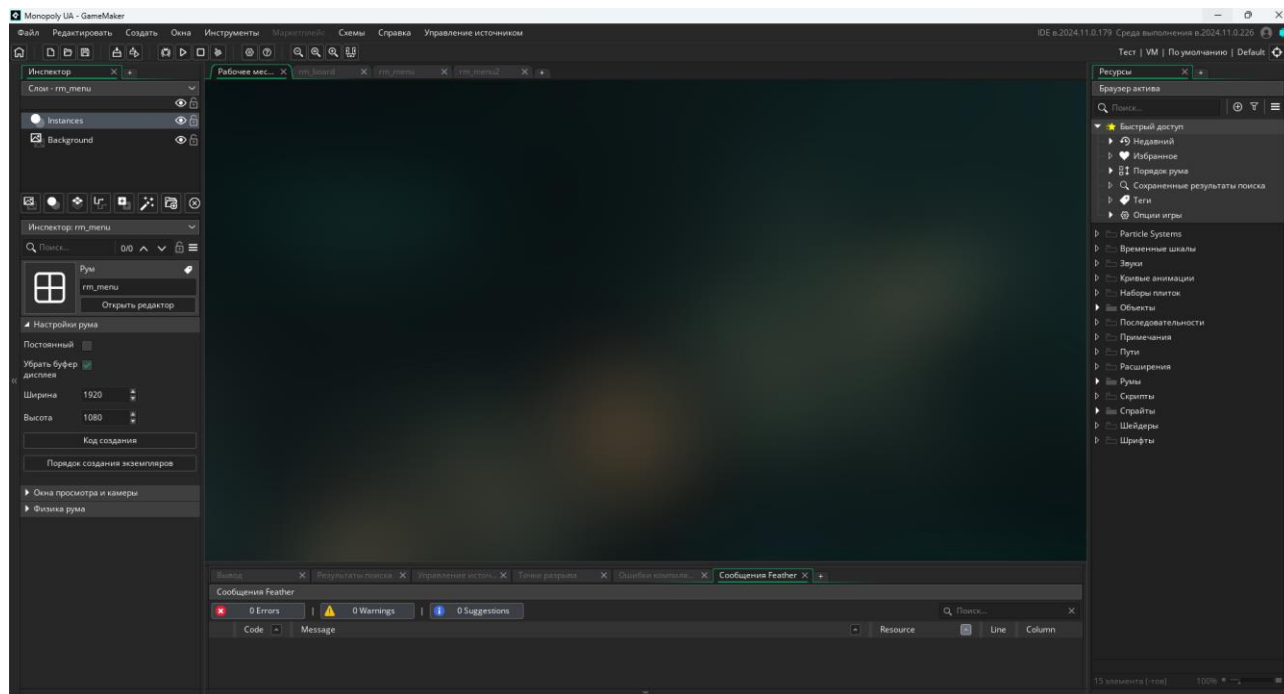


Рис. 1.1. Интерфейс среды Godot

Тим не менш, варто розглянути й інші рушії та проаналізувати їх функціонал, для того щоб вибір був максимально обґрунтованим і об'єктивним. Окрім Godot популярними рушіями на сучасному ринку є, наприклад, Unity, Godot та UnrealEngine.

Unity [37] вважається одним з найпотужніших та найуніверсальніших рушіїв, він має функціонал для реалізації як простих 2D проєктів, так і складних з тривимірною графікою. Проте при розробці саме двовимірної стратегії такий обширний інструментарій є скоріше недоліком, оскільки він не знадобиться під час розробки Монополії, але буде тим не менш ускладнювати і перевантажувати інтерфейс, роблячи його менш зрозумілим для недосвідченого користувача.

Рушій Godot [23] є цікавою альтернативою завдяки своїй легкості, але він все ще поступається Godot за кількістю навчальних матеріалів та бібліотек для швидкої розробки двовимірних ігор.

Що стосується Unreal Engine [38], він розрахований на розробку складних високобюджетних проєктів, відповідно він потребує значного обсягу знань у галузі розробки ігор та потужного апаратного забезпечення, саме тому використання його для створення Монополії є недоцільним.

Підбиваючи підсумок аналізу різних альтернатив серед доступних рушіїв, можна впевнено стверджувати, що середовище розробки GameMaker є найбільш раціональним рішенням. Цей ігровий рушій є золотою серединою, поєднуючи в собі як простоту використання, так і наявність всього необхідного технічного інструментарію для реалізації задуманого проєкту. На відміну від занадто важких професійних рушіїв, GameMaker дозволяє розробнику не витрачати час на налаштування складних графічних систем, а натомість зосередитися на програмуванні ігрового процесу. Наявність зручної мови програмування GML дозволяє створити цікавий та нестандартний ігровий процес який зацікавить будь-якого гравця. Загалом GameMaker надає хороший технічний фундамент для успішного створення і розвитку гри.

1.3. Проєктування ігрової концепції та підготовка ресурсів

Після того, як було визначено мету цієї кваліфікаційної роботи, проаналізовано чималий обсяг корисної літератури та наявні ігрові аналоги, а також обрано ігровий рушій на базі якого проєкт буде реалізовано, наступним кроком є підготовка детального плану розробки гри. На цьому етапі, враховуючи можливості наявного програмного та апаратного забезпечення, потрібно крок за кроком продумати процес створення гри в жанрі стратегія.

Етап планування є надзвичайно важливим, адже без чіткого розуміння мети і які завдання будуть стояти на шляху до неї, процес розробки буде хаотичним і неефективним. Проєктування дозволяє структурувати всі ідеї, що виникли під час вивчення аналогів та перетворити на чіткий план дій, що стане дорожньою картою для подальшої роботи.

На етапі планування варто детально розглянути, що саме являє собою стратегія типу Монополії та як такий проєкт найкраще реалізувати. Далі потрібно створити покрокову інструкцію в якій буде зазначено ключові етапи розробки гри, починаючи зі створення візуальної частини і закінчуючи програмним кодом. Згодом можна буде починати процес розробки відповідно до цієї інструкції.

Настільний варіант Монополії являє собою економічну стратегію для двох і більше гравців. Суть ігрового процесу полягає у пересуванні ігровим полем на певну

кількість клітинок залежно від того, які числа випадуть на гральних кубиках, зазвичай їх два, відповідно діапазон чисел, що може випасти – з 2 до 12. Розмір ігрового поля може бути різним, залежно від конкретного екземпляра гри, але зазвичай на полі приблизно 30 – 40 клітинок. Кожній клітинці на ігровому полі відповідає якась подія, наприклад, найбільше на дошці клітинок власності. Якщо гравець потрапляє на неї, то в нього є можливість викупити цю власність собі за певну суму, потім, коли на цю клітинку потрапить хтось інший, цей гравець повинен буде заплатити певну суму власнику. Якщо купити всі власності в межах одного філіалу (як правило в Монополії клітинки одного філіалу позначаються одним кольором і логічно пов'язані між собою), то далі гравець може будувати там активи і значно підвищувати ту суму, яку платитиме гравець, що стане на цю клітинку. Окрім власності є ще клітинки випадкових подій які можуть, як допомогти, так і зашкодити гравцям які на них потраплять. Також є такі клітинки як “Казино”, “В’язниця” та інші, кожна зі своїми унікальними функціями. По ходу гри гравці які приймали більш оптимальні рішення, або яким просто більше щастило, будуть отримувати перевагу над іншими й зрештою виникне ситуація коли хтось має дуже багато філіалів і грошей, а інші не мають можливості платити за оренду на клітинках на які наступають і стають банкрутами, тобто програють, коли залишиться лише один – гра завершується.

В електронному форматі гра буде мати такий самий ігровий процес, щоправда, кидання кубиків буде відбуватись не вручну, а автоматично, також перерахування коштів буде значно зручніше, оскільки в настільному варіанті їх потрібно фізично рахувати самому, а тут це зробить сама гра. Також у грі буде додано консоль у якій будуть відображатись повідомлення про кожну подію, що сталась на полі.

Тепер, коли є чітке розуміння сутності ігрового продукту який має вийти в результаті, можна створити інструкцію де буде покроково розписано процес створення Монополії.

План розробки гри виглядає наступним чином:

1. Етап – створення візуальних елементів: першим і дуже важливим кроком на шляху створення Монополії є проєктування візуальної частини гри. Це включає розробку гральної дошки, іконок гравців, фішок, а також меню та кнопки в меню. З цією метою доцільно використовувати такі інструменти як, наприклад, Photoshop [7],

Figma [16] тощо. Розпочинати саме зі створення графічної частини є вдалим рішенням, оскільки коли в середовищі рушія розробник починає створювати об'єкти, дуже зручно вже мати перед очима їх візуалізацію, щоб чітко розуміти як вони мають поводитись під час гри.

2. Етап – інтеграція візуальних елементів у ігрові активи: коли зображення фішок, іконок, дошок та меню було створено, потрібно імпортувати їх у середовище GameMaker. Для цього передбачена спеціальна папка під назвою “Sprites”, всередині неї потрібно створити “спрайти”, тобто картинки, для кожного об'єкту що їх потребує, та імпортувати зображення що ми створили раніше. Рекомендується систематизовувати всі елементи всередині папки, щоб вони були не в випадковому порядку, а, наприклад, одна папка була для спрайтів кнопок, друга для спрайтів іконок гравців, третя для спрайтів фону і так далі. Таким чином, коли будуть створюватись об'єкти, буде набагато легше знайти відповідний спрайт.

3. Етап – побудова структури кімнат: на цьому етапі необхідно сформувати послідовність переходу між кімнатами. Щоб реалізувати це в середовищі GameMaker у відповідній папці “Rooms” створюються кімнати у яких гравець буде опинятися у той чи інший момент гри, далі цим кімнатам присвоюються спрайти, тобто фон. Потрібно визначити, у яку з них гравець потрапить при запуску гри, яку кнопку потрібно натиснути та при яких обставинах відбувається перехід в іншу. Згодом, на етапі проєктування кнопок, перехід між кімнатами буде реалізовано функціонально.

4. Етап – створення глобального контролера: глобальний контролер потрібен для того, щоб гра запам'ятовувала деякі дані протягом усієї гри, наприклад це – режим гри, кількість гравців чи глобальні змінні. Він дозволяє всім елементам гри синхронізувати свою логіку відповідно до глобальних параметрів. Створити глобальний контролер у рушії можна створивши відповідний об'єкт у папці “Objects” та надавши йому потрібні властивості.

5. Етап – планування основних ігрових механік: в цей етап входить створення моделі кидка шестигранних кубиків та передавання результату глобальному контролеру, таким чином гравець буде здійснювати хід. Водночас розробляється система руху фішок гравців вздовж певного шляху по клітинках ігрового поля. Фішка

має рухатись на те значення клітинок, яке випало на кубиках, і здійснювати взаємодію з тією клітинкою, на якій зупиниться.

6. Етап – розробка економіки гри: це ключовий етап проєктування Монополії, на ньому створюється система взаємодії гравців з ігровими активами й між собою. Потрібно призначити всім об'єктам власності певну вартість покупки та вартість оренди за різних умов. Також потрібно створити алгоритми купівлі власності які перевіряють, чи має гравець достатньо коштів на балансі, і якщо так, то списує їх. Найскладніше на цьому етапі – це грамотно збалансувати економіку так, щоб одні гравці не мали переваги над іншими й гра залишалась цікавою якомога довше.

7. Етап – Алгоритмізація випадкових подій: в межах цього етапу буде реалізовано алгоритм подій у випадках коли гравець потрапляє на спеціальні клітинки, такі як “Шанс”, “В'язниця”, “Казино”, “Податок” та інші. Потрібно запрограмувати функції для різних клітинок ігрового поля які будуть непередбачуваним чином впливати на ігровий процес. Для реалізації цього потрібно створити генератор випадкових подій який буде обирати дію з масиву можливих сценаріїв та здійснювати її щодо гравця або гравців. Це зробить гру не тільки більш динамічною, а й цікавою, оскільки додасть різноманіття в ігровий процес.

8. Етап – проєктування поведінки ІІІ: оскільки гра передбачає режим гри проти штучного інтелекту, то на цьому етапі буде створено алгоритм який буде імітувати логіку прийняття рішень реальним гравцем. Штучний інтелект повинен аналізувати ситуацію на дошці та на основі цього аналізу приймати рішення. Наприклад, коли він потрапляє на клітинку власності, він має вибір, купляти її чи ні, звичайний гравець в такому випадку має враховувати доцільність покупки саме цієї клітинки враховуючи його грошове становище, оскільки просто викупати всі поля на які стаєш – не краща стратегія, адже тоді не буде грошей на їх розбудову. Тому алгоритм має також оцінити свій баланс, перспективи розвитку саме цього поля власності, і вже тоді вирішувати купляти його чи ні.

9. Етап – інтеграція звукового супроводу: основною метою цього етапу є додавання у гру аудіосупроводу який буде гармонійно доповнювати візуальну частину і підвищувати рівень занурення гравця. Сюди входить, як додавання приємної фонові мелодії що заповнить тишу на ігровій дошці, так і додавання

звукових ефектів для ігрових подій, наприклад кидка кубика, транзакцій валюти тощо. Дизайн звуку це не просто декоративний елемент гри, а важливий елемент UX-дизайну що значно впливає на враження гравців від ігрового процесу.

10. Етап – фінальні налаштування та тестування: завершальний етап розробки присвячений перевірці коректної роботи програмного продукту та, за необхідності, доналаштувань до повної роботоспроможності. В межах цього етапу варто провести тестування гри, щоб виявити, чи не з'являються баги під час гри, чи всі кнопки коректно працюють та відображаються, чи добре налаштовано баланс та поведінку штучного інтелекту. Деякі проблеми мають властивість з'являтися лише у специфічних ситуаціях і для їх виявлення потрібно провести багато тестових ігор, це є дуже важливим для своєчасного усунення недоліків гри. Якщо проблеми буде знайдено, потрібно обережно та точно їх усунути при цьому не зламавши весь алгоритм гри. Коли програмний продукт пройде перевірку тестуванням і буде повністю готовим до гри, тоді процес розробки можна вважати завершеним.

Таким чином, ми маємо готовий план розробки нашого проєкту, завдяки якому є чітке розуміння, яким шляхом ефективніше та простіше рухатись до своєї мети – створення Монополії. Також перед розробкою гри корисно візуалізувати алгоритм функціонування гри, схема має відображати в якому порядку будуть відбуватись події під час гри й від чого ці події будуть залежати. Такий підхід дозволяє побачити структуру проєкту ще до написання коду та уникнути помилок у логіці на старті. Наочне зображення алгоритму стає фундаментом для розробки, де кожна дія гравця має своє чітке місце в системі. Проєктування допомагає структурувати всі ідеї, які виникли під час аналізу ігрових аналогів, і перетворити їх на детальну «дорожню карту». Така візуалізація дає змогу розробнику крок за кроком продумати кожен етап функціонування гри, чітко визначити умови переходу між подіями та зрозуміти, як ігрові механіки залежатимуть одна від одної.

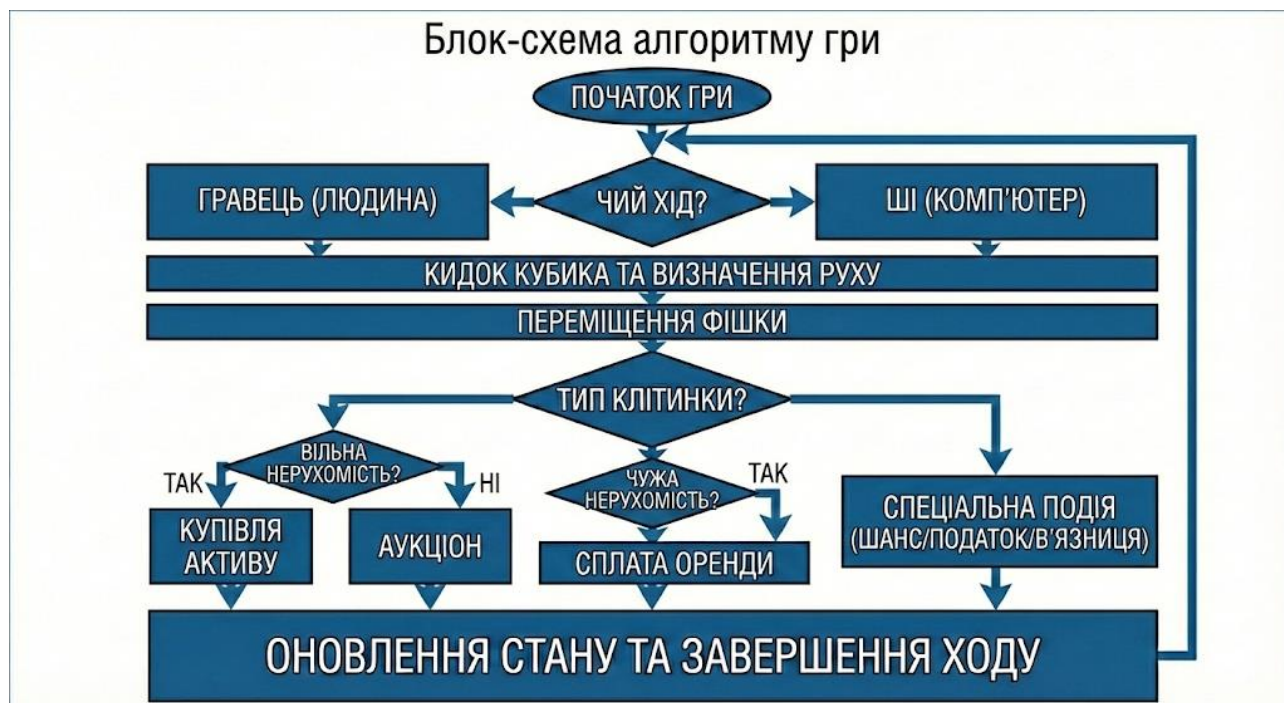


Рис. 1.2. Схематичне зображення алгоритму гри

Як можна побачити за схемою, алгоритм гри є доволі простим. Коли гра починається перший гравець отримує право ходу, хід здійснюється за допомогою умовного кидка двох гральних кубиків, по суті просто генерування випадкового числа від 2 до 12, потім фішка пересувається на відповідну кількість клітинок по ігровому полю. Залежно від того, на яку саме клітинку стала фішка, відбувається певна взаємодія гравця з ігровим середовищем, це може бути покупка філіалу на клітинках оренди, сплата оренди на власності іншого гравця, випробування своєї удачі в казино або пропуск ходу у в'язниці. Коли хід гравця завершено, черга переходить до наступного, і так по колу допоки у грі не залишиться лише один.

Таким чином процес проектування ігрової концепції є завершеним, було створено детальний план розробки ігрового продукту і чітко визначено очікуваний кінцевий результат. Проте, перед тим як переходити до безпосередньо процесу розробки Монополії, варто також проаналізувати всі ресурси, що ми маємо у розпорядженні, це є важливим, оскільки так ми можемо бути впевнені, що маємо весь необхідний інструментарій для реалізації запланованого проекту.

Щодо програмного забезпечення, окрім самого ігрового рушія GameMaker ми також маємо такі інструменти як: фоторедактор Photoshop та онлайн-ресурс Figma.

Photoshop в основному використовується для створення, редагування чи ретушування растрових графічних зображень, це можуть бути логотипи компаній, зображення фішок, іконки гравців тощо. Можливості цього редактора дали змогу створити всі необхідні графічні елементи гри. Figma, своєю чергою, є дуже зручним інструментом створення прототипування інтерфейсу користувача (UI) та загальної конфігурації ігрового поля. Figma дає змогу скласти всі елементи інтерфейсу в єдину композицію, від якої дуже зручно буде відштовхуватись під час розробки.

Також важливо врахувати й можливості апаратного забезпечення. В умовах непередбачуваних відключень електроенергії потрібно мати на увазі, що хоча середовище GameMaker зазвичай автоматично зберігає всі зміни у проєкті, якщо комп'ютер буде знеструмлено під час запису даних, вони можуть бути спотворені. Тому, за можливості, варто скористатися джерелом безперебійного живлення яке дозволить коректно закінчити роботу за будь-яких умов. Тепер, коли проблему відключень світла враховано, потрібно проаналізувати конфігурацію комп'ютера та зробити висновки щодо його спроможності виконувати такі завдання як розробка Монополії.

Основою системи є процесор AMD Ryzen 5 7500F, цей процесор є дуже хорошим і надійним, він забезпечує високу швидкість та ефективність обробки коду, що є важливим під час створення гри. Відеокарта AMD Radeon RX 7800 XT дозволяє працювати з високою роздільною здатністю в таких середовищах як Photoshop або Figma, а також компілювати графічні ефекти без втрати частоти кадрів. Комп'ютер оснащений оперативною пам'яттю обсягом 32 ГБ що є корисним при одночасному запуску багатьох ресурсозатратних програм. Наостанок, система оснащена SSD накопичувачем на 500 ГБ який є хорошою базою для системних файлів і HDD диском об'ємом у 1 ТБ де є вдосталь місця для всіх можливих медіафайлів чи архівів розробки. Таким чином, у нас в розпорядженні є потужний комп'ютер на якому реалізація даного проєкту є цілком реальним завданням.

РОЗДІЛ 2. ПРАКТИЧНИЙ ЕТАП СТВОРЕННЯ ПРОГРАМНОГО ПРОДУКТУ

2.1. Розробка графічних ресурсів та архітектури об'єктів у середовищі GameMaker

Згідно з планом розробки нашої гри, першим етапом цього процесу є створення візуальних елементів. На перший погляд, таке рішення може здаватись нелогічним, оскільки частіше в першу чергу під час розробки програмного продукту створюють саме функціональну частину, але знаючи особливості функціонування обраного ігрового рушія, а також специфіку жанру відеогри, причини такого рішення стають очевидними [5]. Ігровий процес Монополії практично повністю складається з простих натискань на кнопки інтерфейсу, тоді як складні механіки руху гравця та взаємодії з ігровим всесвітом, як це буває у 3D – іграх, відсутні. Це означає, що елементи інтерфейсу, такі як кнопки, іконки фішок та іконки гравців – це не другорядні аспекти гри які реалізуються ближче до кінця гри, а основна і визначальна частина нашого продукту. Коли графічні елементи буде створено, ми матимемо можливість розмістити їх у середовищі рушія та побачити повноцінну картину, яку вони утворюють, і на основі цього вже буде зручніше розробляти функціональну частину гри.

Перед початком створення візуальних елементів потрібно визначити, які саме частини інтерфейсу нам потрібно створити окремо, а які буде зручніше реалізувати пізніше за допомогою коду в GameMaker. Також необхідно обрати графічний редактор, за допомогою якого ці елементи буде створено. Для розробки інтерфейсу найбільш зручним є онлайн-сервіс Figma, він має весь необхідний інструментарій для створення візуальних елементів ігор чи сайтів, а також, що не менш важливо, є простим та інтуїтивно зрозумілим.

В першу чергу варто створити меню гри, це саме те місце куди гравець потрапляє одразу при її запуску. Меню гри має бути простим, не перевантаженим інформацією, але водночас красивим та інтригуючим. Назва, яку матиме наша гра – це “Monopoly UA”, значить, меню має бути на відповідну тематику. Гарною ідеєю буде додати на фон панорамне зображення якого-небудь українського міста, наприклад – Києва. Оскільки такі зображення, як правило, досить контрастні, його

слід трохи приглушити та затемнити, щоб яскраві надписи на його фоні було краще видно [27]. Отримати такий ефект можна досить просто, рівно позаду зображення потрібно помістити прямокутник темного кольору, а самому зображенню зменшити показник непрозорості приблизно до 80%. Тепер, коли фон готовий, залишається лише додати назву гри та, бажано, інформацію про розробника. Назву варто розмістити по центру вгорі та написати великим цікавим шрифтом, можна також проєкспериментувати з кольоровим заповненням і тіннями. Згадку про розробника потрібно зробити читабельною, але розмістити її краще десь у куточку, наприклад внизу справа, оскільки ця інформація є другорядною для користувача.

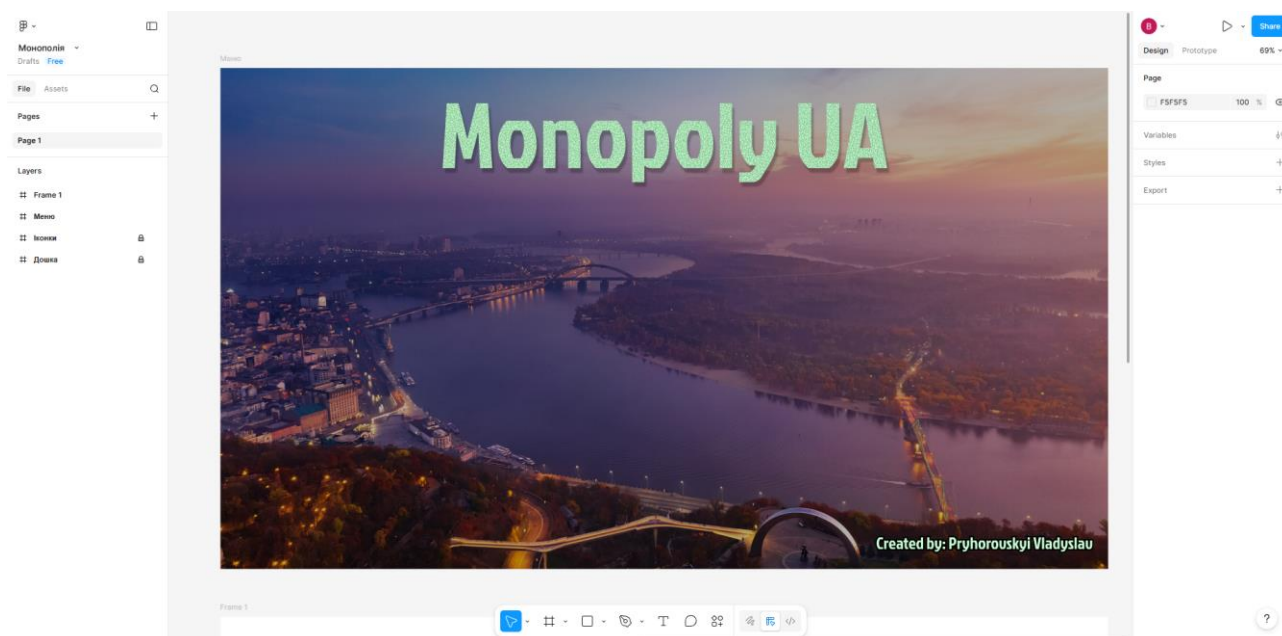


Рис. 2.1. Вигляд меню гри

Звісно, у меню ще потрібно буде додати кнопки якими гравець буде починати гру, але основу меню можна вважати створеною. Наступним кроком, логічно, йде створення цих кнопок на які гравець буде натискати у меню.

Для того, щоб розуміти, які саме кнопки для меню потрібно створити, потрібно знати, які режими гри будуть присутні у грі. Згідно з планом, в “Monopoly UA” у її першій ітерації матимуть місце два режими гри – локальна гра та гра проти штучного інтелекту. У перспективі є можливість додавання режиму гри в онлайн, коли користувачі з різних пристроїв зможуть грати разом, але поки що цей режим реалізовано не буде.

Таким чином нам потрібно три кнопки в меню: “Локальна гра”, “Гра проти ШІ” та “Онлайн-гра”. Кнопки мають бути досить великими відносно вікна меню, яскравими та інформативними. Наприклад, основне забарвлення кнопок можна зробити зеленим і додати чорну обводку, їх форму краще зробити прямокутником з трошки заокругленими кутами, так кнопки виглядають гармонійно. У кнопки онлайн-гри колір варто змінити на сірий для того, щоб було очевидно, що вона не клікабельна. На кожную з кнопок потрібно додати відповідний текст та іконку, таким чином вони будуть більш інтуїтивно зрозумілими для гравців.

Коли кнопки вибору режиму гри буде створено, потрібно також зробити кнопки для вибору кількості гравців. Зазвичай монополія розрахована на 2-4 гравців, тобто нам потрібні ще три кнопки – “два”, “три” і “чотири”.



Рис. 2.2. Кнопки в меню гри

Таким чином, ми маємо готові кнопки які буде розміщено у меню і на які гравець буде натискати коли запустить відеогру. Опціонально, можна додати написи з поясненнями під кнопками, це не є обов’язковим оскільки функціонал цих кнопок є очевидним, але для покращення інтерфейсу – цілком доцільно.

Тепер ми маємо всі необхідні елементи меню, наступним кроком є розробка гальної дошки. Приблизна конфігурація ігрової дошки в монополії являє собою квадрат або прямокутник у зовнішній частині якого знаходяться поля по яких гравці

пересуваються. Оскільки стандартне співвідношення сторін сучасного монітора складає 16 на 9, або 1920 на 1080 пікселів, краще створити поле у вигляді прямокутника який буде займати повний екран.

У настільному варіанті цієї гри клітинки по яких ходять гравці знаходяться по периметру дошки, а в центрі розташовані, зазвичай, карточки випадкових подій та зображення казино. Але в цифровому варіанті гри потреба у них відпадає, адже весь необхідний функціонал обчислюється комп'ютером та відбувається автоматично. З іншого боку, оскільки дошка займає весь екран, в центрі потрібно помістити інформацію про кількість гравців та їх поточний баланс, також для кращого розуміння ходу гри корисно буде створити консоль у якій будуть відображатись останні події у грі [12].

Розпочати процес створення ігрової дошки варто з розмітки секторів під клітинки. Спочатку всередині великого прямокутника що являє собою всю дошку потрібно створити прямокутник менше, він буде відокремлювати неігрову зону, тобто центр дошки, з територією клітинок. На фон центру поля можна, аналогічно до меню, додати красиве зображення що відповідає тематиці гри. Далі зону що залишилась по периметру потрібно поділити на певну кількість клітинок, приблизно від 30 до 40. Кожній з клітинок варто додати іконку яка буде відображати її суть. Окремо для клітинок власності краще зробити ще позначки по кольорах, які будуть означати що певні клітинки відносяться до однієї групи, це дуже важливо для ігрового процесу монополії.

Щодо центру дошки, там, як вже було зазначено, потрібно передбачити інтерфейс для гравців, а саме показники їх балансу та консоль. Для того щоб іконки гравців та повідомлення консолі було краще видно, краще зробити спеціально слоти для них, наприклад, темні прямокутники з відсотком непрозорості близьким до 100. Далі можна розробляти самі іконки гравців, а також іконки фішок, які будуть пересуватись по клітинках. Важливо пам'ятати, що додавати зображення іконок на дошку на цьому етапі поки що не потрібно, оскільки їх кількість може змінюватись залежно від кількості гравців, тому у середовищі Figma їх варто розмістити окремо від дошки й потім додавати в гру у вигляді окремих сутностей [35]. Оскільки в грі

передбачено до чотирьох гравців, іконки гравців і фішок будуть являти собою зображення в чотирьох кольорах – зелений, синій, червоний та жовтий.

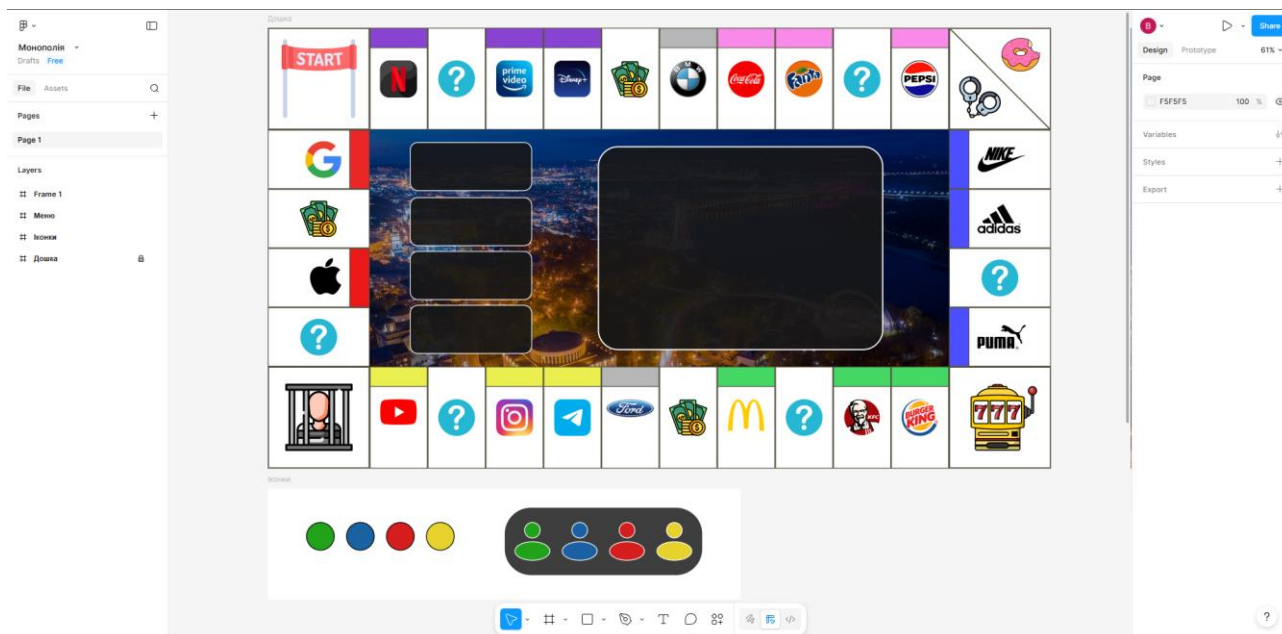


Рис. 2.3. Дошка та іконки гравців

Таким чином, створення візуальних елементів гри в основному завершено. Звісно, по ходу гри гравцю доведеться натискати на ті чи інші кнопки, які також вимагають візуалізації, але їх створення зручніше відкласти до етапу програмування ігрового процесу, оскільки вони не є постійною частиною інтерфейсу, а з'являються лише у певних ситуаціях.

Тепер, коли ми маємо готовий набір візуальних елементів, можна перейти до етапу проєктування архітектури об'єктів у середовищі GameMaker. Цей етап є надзвичайно важливим, адже вдало розроблена архітектура значно прискорить та спростить процес створення гри.

Процес перенесення статичних графічних активів у динамічне програмне середовище вимагає чіткої ієрархії та систематизації всіх наявних ресурсів. Для того щоб перетворити набір зображень на цілісний ігровий світ, необхідно спочатку сформуванати логічний простір, у якому відбуватимуться всі майбутні події. Це дозволяє структурувати роботу об'єктів і забезпечити плавний перехід між різними станами програми.

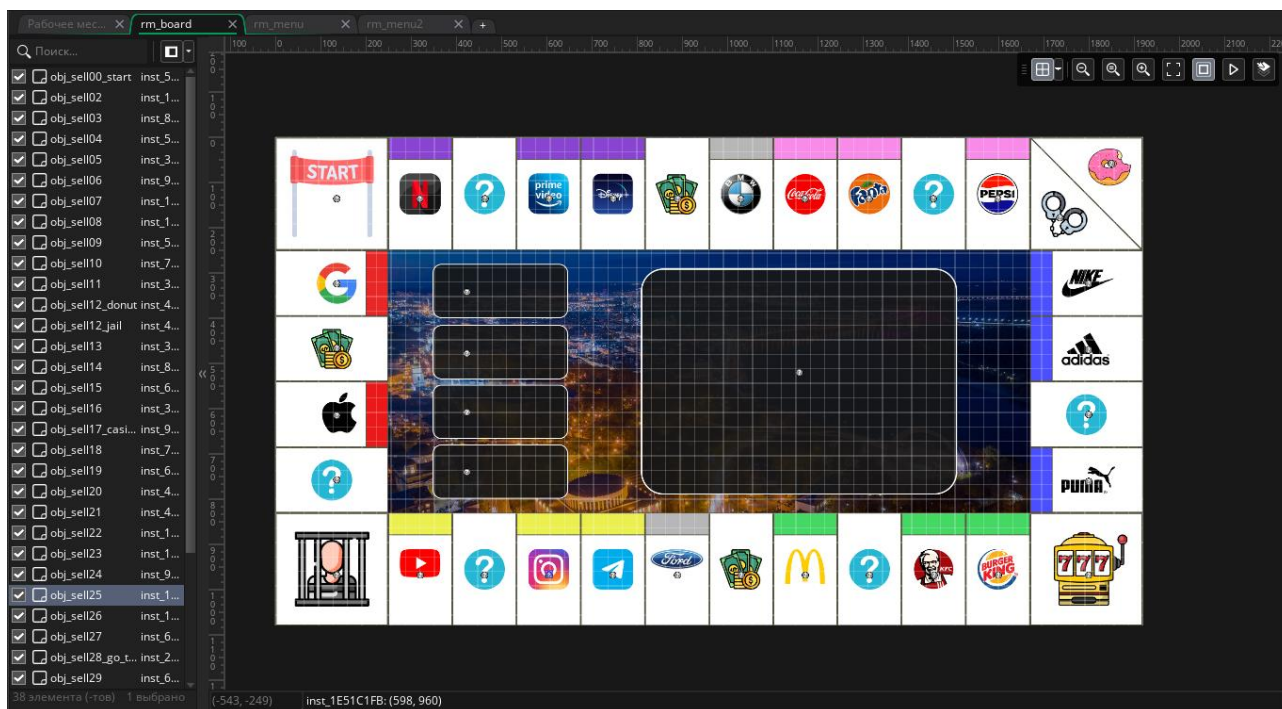


Рис. 2.4. Створення румів

Першим кроком на цьому шляху є створення румів, тобто кімнат, вони являють собою середовище, в яке гравець потрапляє при запуску гри, або за певних обставин. Виходячи з логіки ігрового процесу, для нашої монополії потрібно створити три рума. Перший – головне меню яке вже було розроблене на попередньому етапі, там будуть розташовані кнопки вибору режиму гри, цей рум необхідно зробити головним, щоб при запуску гравець потрапляв саме сюди. Другий – також меню, але цього разу замість вибору режиму там буде вибір кількості гравців, третій – гральна дошка, це вже місце де буде відбуватись сама гра.

Тепер, коли ми маємо всі потрібні руми та їх ієрархію, можна переходити до створення об'єктів та спрайтів. Об'єкт – це сутність якій притаманні певні властивості, розмір, швидкість, поведінка при натисканні або наведенні мишкою тощо, спрайт, своєю чергою, це фактично зображення об'єкта. Потрібно зробити каталог об'єктів для всіх сутностей що будуть мати місце в грі. В середовищі GameMaker для цього є панель ресурсів, там можна знайти папку яка називається “Об’єкти”, саме туди і слід їх додати. Рекомендується, для того, щоб структурувати розташування об'єктів, згрупувати їх за певними ознаками, наприклад за призначенням або розташуванням у тому чи іншому румі. Окрім очевидних об'єктів,

таких як кнопки, фішки чи клітинки на полі, потрібно створити особливі об'єкти, а саме – контролер гри, консоль та батьківську клітинку, їх важливість стане очевидною пізніше по ходу розробки відеогри. Тим об'єктам, яким це потрібно, потрібно призначити відповідний спрайт, в першу чергу це елементи інтерфейсу, як в меню, так і на дошці, також спрайти потрібно призначити до кожного руму.

Коли створення об'єктів та спрайтів буде завершено, ми матимемо зручний каталог всіх основних елементів монополії та зможемо редагувати їх властивості, таким чином налаштовуючи ігровий процес.

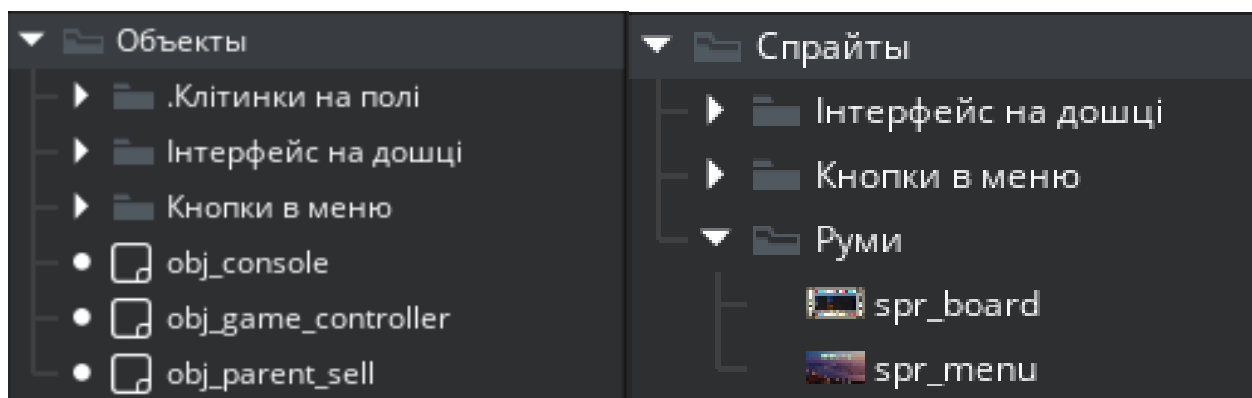


Рис. 2.5. Об'єкти та спрайти

Окрім румів, об'єктів та спрайтів, важливими елементами гри є також: звуки, які покращують користувацький досвід, скрипти, які регулюють те, яким чином у грі відбуваються ті чи інші події, шрифти, та інші елементи архітектури. Поки що створювати їх рано, але на певному етапі розробки вони також стануть невіддільною частиною структури гри в середовищі рушія.

2.2. Реалізація ключових ігрових механік та логіки взаємодії

Тепер, коли ми маємо всі необхідні об'єкти та їх спрайти в системі рушія, можна починати реалізовувати безпосередньо ігровий процес монополії.

Першим кроком є реалізація переходу між румами при натисканні на кнопки, саме тому було важливо не намалювати кнопки у вигляді звичайного зображення, а створити їх окремо та додати в якості об'єкта в GameMaker. Реалізувати сам перехід з рума в інший рум при натисканні на кнопку є доволі просто задачею, для цього у події Step (крок) об'єктів кнопок додається код який змушує гру перейти до

потрібного руму при натисканні лівою кнопкою миші. Але мало просто перейти в інший рум, також важливо, щоб гра запам'ятала, який саме режим гри та яку кількість гравців було виставлено, аби розуміти, скільки фішок створювати на дошці та ким вони будуть керуватись, гравцем чи штучним інтелектом.

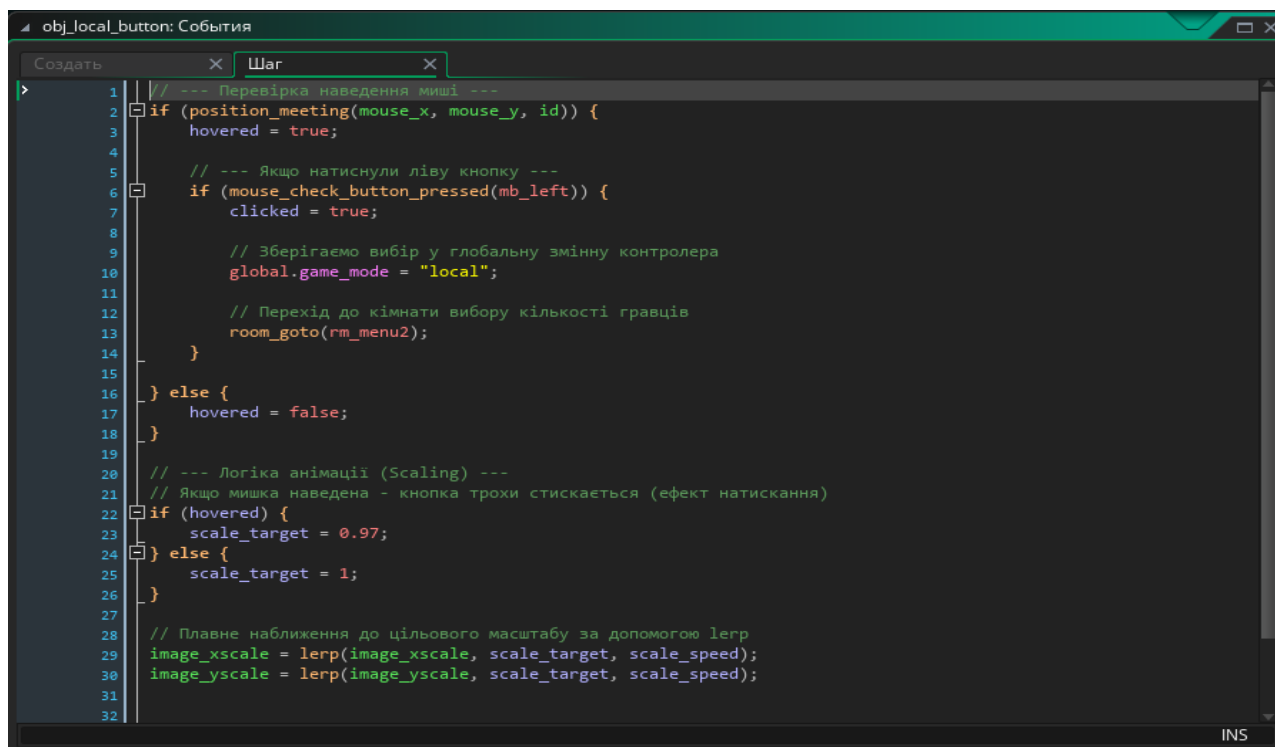


Рис. 2.6. Код для однієї з кнопок

Для цієї задачі знадобиться створений раніше `obj_game_controller` (контролер гри), цей об'єкт, в тому числі буде містити інформацію про актуальний режим гри та кількість гравців, а в коді кнопок потрібно лише запрограмувати внесення змін до цих параметрів гри. Наостанок, для покращення візуального досвіду, варто анімувати ці кнопки, наприклад зробити щоб при наведенні миші вона трохи зменшувалась. Тепер ми маємо працюючий алгоритм переходу між кімнатами, працює він наступним чином: при запуску гри ми опиняємось у меню з двома кнопками – локальна гра та гра проти штучного інтелекту, також є кнопка онлайн гри, але вона неклікабельна, натиснувши одну з кнопок ми переходимо до вибору кількості гравців і вже потім, коли ми встановили кількість гравців, гра переміщає нас безпосередньо на ігрову дошку, паралельно запам'ятовуючи параметри які ми встановили.

Тепер ми готові переходити до проєктування основи ігрового процесу. Найголовнішим елементом гральної дошки є ігрові клітинки, саме на них будуть

розміщуватись фішки гравців. Оскільки на спрайті дошки положення клітинок є цілком очевидним, розмістити їх об'єкти зараз не стане проблемою. На попередньому етапі, коли була розроблена архітектура гри, в папку “об'єкти” вже було додано об'єкти всіх клітинок, зараз потрібно лише встановити їх у відповідних місцях у румі `rm_board` (гральна дошка). Починаючи з клітинки “старт” ми розміщуємо кожен об'єкт на його місце, коригувати положення об'єкта можна за допомогою спеціального вікна, де встановлюються параметри об'єкта, в тому числі його координати у румі. Таким чином ми йдемо клітинка за клітинкою, допоки на кожній з них не буде встановлено відповідний об'єкт. Створення об'єктів клітинок є дуже важливим, оскільки вони є опорними точками, до яких будуть рухатись фішки під час ходу, а також саме цим об'єктам будуть надаватись властивості які визначатимуть їх тип і вартість покупки чи оренди.

Наступним кроком варто створити об'єкти які являють собою слоти, тобто місця, де будуть знаходитись іконки гравців та ігрова консоль. Чотири об'єкти іконок гравців розміщуються у чотирьох прямокутниках, що знаходяться з лівої сторони дошки, на цих місцях будуть з'являтись іконки гравців, саме така кількість, яку користувач вибере в меню гри. У великому прямокутнику, що знаходиться справа, встановлюється об'єкт консолі, тут по ходу гри будуть з'являтись повідомлення що інформуватимуть гравців про всі події на дошці. Поки що складно передбачити, як саме краще розмістити цей об'єкт, оскільки текст може бути різних розмірів, але цей параметр завжди можна налаштувати при необхідності. Таким чином, ми маємо рум дошки у якому розміщено всі об'єкти, які необхідні для коректного функціонування ігрового процесу.

Завершення технічного облаштування ігрової сцени дозволяє перейти від фази статичного проєктування до етапу програмної активації ігрового середовища. Цей перехід передбачає встановлення зв'язку між графічними компонентами та глобальним контролером, який керує станом усієї системи. Саме наявність підготовлених слотів та клітинок дає змогу програмно ініціалізувати початковий стан сесії та наповнити її активними сутностями.

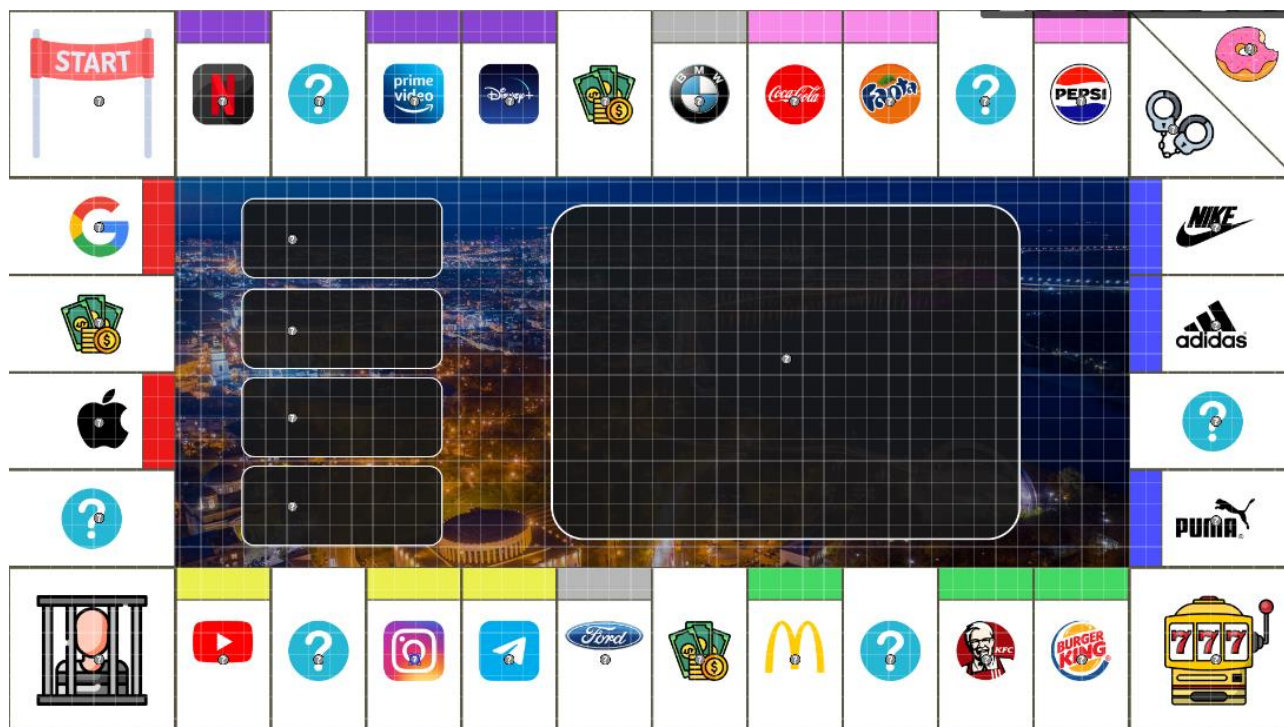


Рис. 2.7. Ігрові клітинки та слоти для іконок і консолі

Тепер, коли всі клітинки та слоти мають відповідні об'єкти, можна починати розробку безпосередньо ігрового процесу монополії. Для початку гри потрібно створити на полі певну кількість фішок гравців, саме таку, яку обрав користувач у меню. Оскільки значення кількості гравців вже записано в базу даних контролера, тепер потрібно написати код, також у контролері гри, який буде ставити на полі відповідну кількість фішок.

У події “крок” об'єкта контролера гри потрібно прописати код який буде виконувати дві задачі. Перша – створення іконок гравців у спеціальних слотах в середині дошки, вони є не тільки елементом дизайну, який покращує розуміння подій в грі, а ще й важливою опорною точкою для кнопок та лічильників які буде створено пізніше та які матимуть ключове значення для гри. Друга – створення фішок гравців на стартовій клітинці, коли гра розпочнеться, ці клітинки будуть пересуватись полем та взаємодіяти з дошкою. Оскільки всі фішки прив'язані до однієї точки, в об'єкті кожної з фішок варто прописати певний відступ, наприклад вгору чи вниз, і у кожній з них цей показник повинен відрізнятись, таким чином опинившись на одній клітинці вони не будуть повністю перекривати одна одну.

Наступним етапом є розробка механіки руху фішок. В першу чергу програмується умовний кидок кубика, а точніше двох кубиків. Для програмування процесу кидка кубика варто створити спеціальний скрипт який буде відповідати за цю дію. В середовищі GameMaker є папка, що називається “Scripts”, саме там його слід створити та назвати, наприклад, `scr_perform_roll` (скрипт виконати кидок).

Алгоритм роботи цього скрипту виглядає наступним чином:

1. Перевіряється, чи гравець який має кидати кубик не робив цього раніше в межах цього ходу, це забезпечує, щоб здійснити кидок можна було лише раз за один хід.

2. Здійснюється умовний кидок кубиків, по суті просто генерується випадкове число від 2 до 12, яке являє собою кількість клітинок на яку фішка переміститься вперед.

3. Визначається, яка саме фішка зараз ходить, для цього скрипт звертається до змінної “`global.current_turn`” та визначає, якій фішці передати команду до руху, також скрипт відправляє текстове повідомлення яке відображається у консолі.

4. Здійснення руху: знаючи яка фішка повинна ходити та як далеко, скрипт передає об’єкту фішки певне значення для змінної “`steps_to_move`”, яку попередньо було створено в коді всіх фішок, таким чином вона знає скільки кроків вперед має зробити. Також, для того, щоб хід виглядав плавно, додається затримка перед кожним кроком, таким чином фішка не телепортується до місця призначення миттєво, а йде поступово.

Важливо зазначити, що для того щоб цей скрипт працював, потрібно кожному з об’єктів клітинок присвоїти певний індекс, починаючи від “старт” з індексом “0”, наступна з індексом “1”, і так далі по черзі, таким чином гра буде розуміти яка клітинка слідує по черзі за якою і хід буде здійснено коректно.

Коли логіка руху та система індексації готові, ми маємо робочу основу для гри. Проте зараз цей алгоритм існує лише у вигляді коду і ніяк не взаємодіє з людиною. Щоб гравець міг сам запускати процес і робити хід, потрібно додати візуальний елемент керування. Тому наступним кроком є створення кнопки, яка з’єднає дії гравця із нашими скриптами.

```

1 function scr_perform_roll() {
2     if (!global.can_roll) exit;
3     global.can_roll = false;
4
5     var d1 = irandom_range(1, 6);
6     var d2 = irandom_range(1, 6);
7     var total = d1 + d2;
8
9     // Логи
10    var p_idx = global.current_turn - 1;
11    var p_names = ["Blue", "Green", "Red", "Yellow"];
12    // Переконайся, що функція логу існує і працює
13    if (variable_instance_exists(obj_game_controller, "scr_add_log")) {
14        obj_game_controller.scr_add_log(p_names[p_idx] + " rolled: " + string(total));
15    }
16
17    // Знаходимо фішку
18    var p_obj = noone;
19    switch(global.current_turn) {
20        case 1: p_obj = obj_blue_chip; break;
21        case 2: p_obj = obj_green_chip; break;
22        case 3: p_obj = obj_red_chip; break;
23        case 4: p_obj = obj_yellow_chip; break;
24    }
25
26    if (instance_exists(p_obj)) {
27        // --- ВИПРАВЛЕННЯ ТУТ ---
28        // Ми просто передаємо кількість кроків.
29        // Фішка сама розбереться з координатами в Step Event.
30        p_obj.steps_to_move = total;
31
32        // Таймер для старту (щоб не стрибнула миттєво)

```

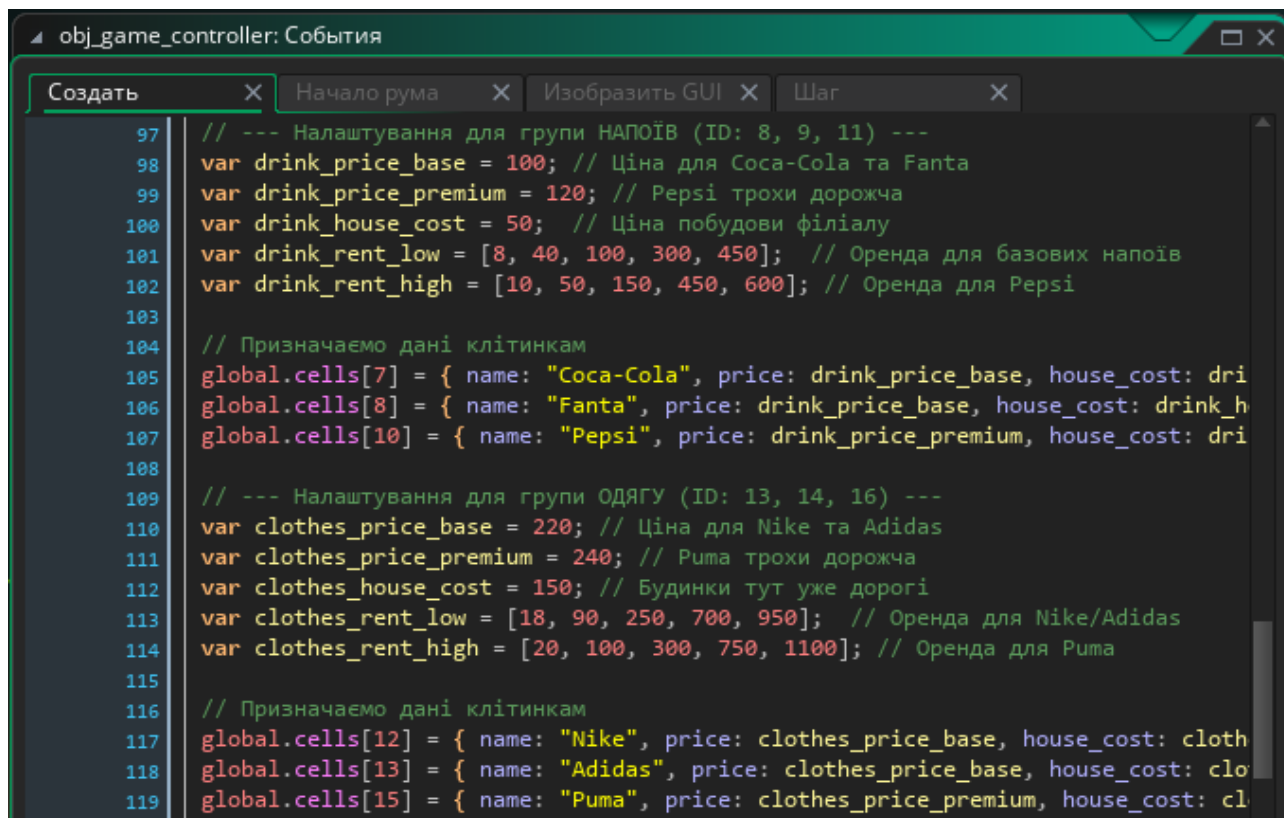
Рис. 2.8. Скрипт кидка кубиків

Тепер ми маємо повністю налаштований процес ходу гравця, але потрібно також подумати про те, як він буде викликатись, адже на цей момент не має жодного механізму запуску цього скрипта. Найпростіший спосіб це створювати кнопку за допомогою події “малювання” об’єкта контролера гри, задавши їй певні розміри, колір, текстове наповнення та просту анімацію при наведенні миші, а в події “крок” запрограмувати запуск скрипта при натисканні на кнопку лівою клавішею миші. Таким чином алгоритм кидка кубика є готовим.

Наступним кроком у розробці монополії є присвоєння вартостей клітинкам власності та реалізація механіки їх покупки. Купівля та розбудова власності є основою ігрового процесу монополії, те наскільки оптимально гравцю вдасться витратити свої ресурси на прибуткову власність є визначальним фактором у результаті гри.

Першим кроком у створенні цієї непростой системи є присвоєння клітинкам їх вартості. Як можна зрозуміти зі спрайту дошки, клітинки власності на полі розподілені на декілька груп у яких філіали логічно пов'язані між собою, наприклад

група виробників одягу, напоїв або месенджерів. У події “створення” контролера потрібно прописати ціну покупки клітинки, ціну побудови філіалу на ній, а також ціну оренди яку платитиме той, кому не пощастило на неї стати. Для різноманіття, можна навіть у межах однієї групи присвоїти різним клітинкам різну вартість, це типове явище для монополії.



```

97 // --- Налаштування для групи НАПОЇВ (ID: 8, 9, 11) ---
98 var drink_price_base = 100; // Ціна для Coca-Cola та Fanta
99 var drink_price_premium = 120; // Pepsi трохи дорожча
100 var drink_house_cost = 50; // Ціна побудови філіалу
101 var drink_rent_low = [8, 40, 100, 300, 450]; // Оренда для базових напоїв
102 var drink_rent_high = [10, 50, 150, 450, 600]; // Оренда для Pepsi
103
104 // Призначаємо дані клітинкам
105 global.cells[7] = { name: "Coca-Cola", price: drink_price_base, house_cost: dri
106 global.cells[8] = { name: "Fanta", price: drink_price_base, house_cost: drink_h
107 global.cells[10] = { name: "Pepsi", price: drink_price_premium, house_cost: dri
108
109 // --- Налаштування для групи ОДЯГУ (ID: 13, 14, 16) ---
110 var clothes_price_base = 220; // Ціна для Nike та Adidas
111 var clothes_price_premium = 240; // Puma трохи дорожча
112 var clothes_house_cost = 150; // Будинки тут уже дорогі
113 var clothes_rent_low = [18, 90, 250, 700, 950]; // Оренда для Nike/Adidas
114 var clothes_rent_high = [20, 100, 300, 750, 1100]; // Оренда для Puma
115
116 // Призначаємо дані клітинкам
117 global.cells[12] = { name: "Nike", price: clothes_price_base, house_cost: cloth
118 global.cells[13] = { name: "Adidas", price: clothes_price_base, house_cost: clo
119 global.cells[15] = { name: "Puma", price: clothes_price_premium, house_cost: cl

```

Рис. 2.9. Присвоєння вартості клітинкам

Після того як всім полям власності було виставлено ціну, потрібно також подбати, щоб вона відображалась в інтерфейсі користувача, інакше він не буде розуміти, скільки потрібно платити за той чи інший філіал. Реалізувати це можна за допомогою події “малювання” в об’єкті клітинки, але прописувати код для кожного окремо – довго та неефективно. Для розв’язання цієї задачі варто створити спеціальний об’єкт – материнську клітинку, в коді якої програмується відображення вартість кожної клітинки, цей об’єкт потрібно вказати як батьківський для всіх клітинок, таким чином у кожній з них буде відображатись саме їх ціна.

Також варто звернути увагу, що не всі клітинки знаходяться в однаковому положенні, деякі стоять вертикально, деякі – горизонтально. Враховуючи це, клітинки потрібно розділити на три групи: вертикальні, горизонтальні зліва та

горизонтальні справа, для кожної з груп потрібно прописати різні координати відображення вартості. Для тих, що розташовані вертикально, текст краще зсунути вниз відносно центру, майже до нижньої межі клітинки, для цього там вже заздалегідь передбачене вільне місце. Для тих, що стоять горизонтально справа та зліва, до правого та лівого краю відповідно. Також для горизонтальних клітинок текст варто нахилити так, щоб він був повернутий основою до краю дошки, бо якщо залишити його горизонтально, він може наїхати на логотип компанії.

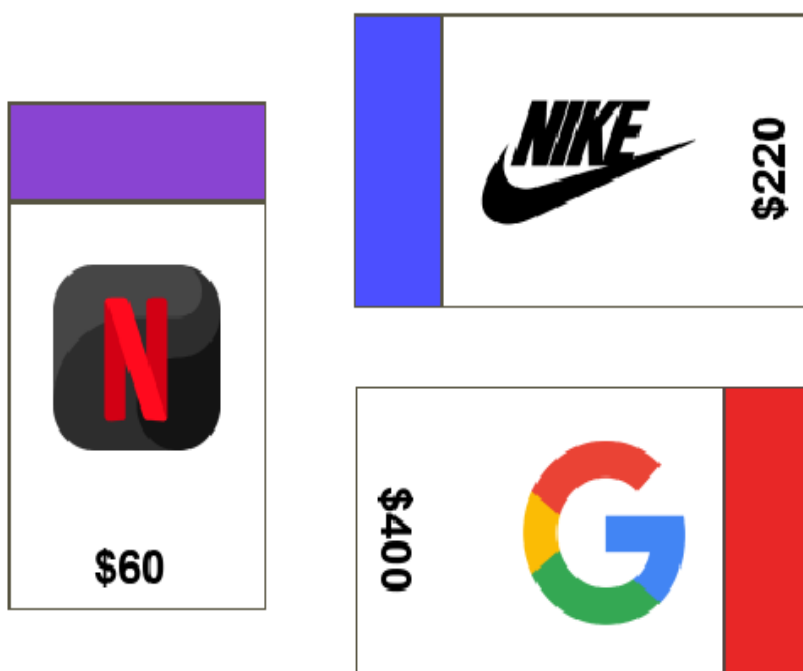


Рис. 2.10. Відображення цін на клітинках

На даному етапі ми маємо готовий алгоритм здійснення ходу, а також визначені вартості клітинок власності, наступним етапом розробки, логічно, є програмування системи здійснення покупки. Ця задача є непростю, оскільки вимагає комплексного рішення одразу багатьох задач.

Коли гравець кидає кубик і опиняється на клітинці власності яка нікому не належить, перед ним постає вибір – купити або залишити. Для того щоб гравець міг реалізувати свій вибір, в інтерфейсі гри мають з’являтися відповідні кнопки – “Buy” та “Pass”. Щоб ця система працювала, у події “крок” об’єкта контролера прописується спеціальний програмний код, він перевіряє, чи знаходиться зараз гра у стані прийняття рішення, якщо так, то біля іконки відповідного гравця створюються дві

кнопки – “купити” та “залишити”. Якщо гравець натискає “Buy”, то, якщо гравець має достатньо коштів, власність переходить до нього, і замість ціни покупки на клітинці вже буде вказуватись ціна оренди, яку платитимуть інші гравці. Якщо гравець вирішив не купувати власність і натискає “Pass” то нічого не відбувається і хід переходить до наступного.



Рис. 2.11. Стан прийняття рішення

Важливо також додати візуальне відображення того, кому саме належить власність. Реалізувати це можна досить просто – коли та чи інша клітинка викуповується гравцем, потрібно фарбувати її в колір відповідний кольору його фішки. Тут знадобиться створений раніше об’єкт батьківської клітинки, у події малювання якого, на додачу до ціни, потрібно написати код який буде створювати прямокутник зеленого, синього, червоного або жовтого кольору, залежно від власника. Для того щоб створений прямокутник не закривав повністю зображення клітинки, йому слід виставити рівень непрозорості приблизно на 20%, таким чином буде чітко видно як колір власника, так і зміст клітинки [2]. Аналогічно з відображенням цін, розміщення прямокутника буде відрізнятись у клітинок зліва, справа, вгорі та внизу, тому тут також потрібно поділити клітинки на групи за цим параметром і кожній з груп прописати свої координати, так щоб прямокутник рівно ставав поверх них.

Для успішної реалізації системи покупок власності обов’язково потрібно запрограмувати систему нарахування та списання коштів гравцям. У контролері гри, в події “створення” визначається початкова сума з якою гравці починають, а в коді фішок прописується, наприклад, нарахування додаткових коштів за проходження

кола. Сума яку гра має списати за покупку власності або оренду вже була прописана раніше, тому зараз програмувати це знову не потрібно.

Тепер, коли алгоритм викупу власності працює, є можливість перейти до наступного кроку – програмування системи розбудови філіалів. Реалізація цього елементу ігрового процесу є досить складним завданням, оскільки потребує неабияких навичок програмування та розуміння логіки подій у грі. Суть цієї механіки є приблизно однаковою у всіх іграх жанру монополії та виглядає наступним чином: якщо гравець оволодіває всіма клітинками власності в межах однієї групи, то він має можливість будувати на них філіали, в результаті чого гравці, які потрапляють на відповідну клітинку, платитимуть власнику значно більшу суму.

Щоб реалізувати цю механіку на практиці, потрібно запрограмувати відповідні події у контролері гри, а також в об'єкті материнської клітинки [6]. В першу чергу необхідно пересвідчитись що клітинки власності об'єднанні в групи не лише візуально на дошці, а й у коді їх створення, інакше алгоритм не буде розуміти, які саме з них утворюють філіал. Далі у події “крок” контролера програмується функціональна основа системи. Якщо всі клітинки однієї групи належать одному гравцю, то на початку свого ходу, тобто до кидка кубика, він має право побудувати або продати філіал. Щоб побудувати або продати філіал, гравцю потрібно натиснути лівою клавішею миші на ту клітинку, з якою він хоче взаємодіяти, після натискання з'являться три кнопки. Кнопка “Build” – це побудова філіалу, якщо користувач натискає на неї, то з його рахунку списується сума яка відповідає вартості побудови філіалу на цій клітинці, а ціна оренди для інших гравців, відповідно, також зростає. Кнопка “Sell” відповідає за продаж філіалу, це може бути необхідним кроком якщо у гравця немає грошей для сплати штрафу або оренди, проте при продажу гравцю повертається лише половина від ціни покупки, таким чином він має більш обдуману приймати рішення про побудову філіалу. Аналогічним чином цією кнопкою можна продати й саму клітинку тим самим зробивши її знову нейтральною. Наостанок кнопка “Close” просто закриває це меню, її гравець натискає тоді коли всі необхідні дії було виконано.

Коли всі вищезазначені функції буде реалізовано, потрібно подбати про регулювання можливості побудови або продажу філіалу, а саме, необхідно прописати

що гравець не може будувати багато філіалів на одній клітинці, а на іншій – не будувати зовсім, тобто робити це потрібно рівномірно, те саме стосується і продажу. Для реалізації цього потрібно просто встановити певну умову, при побудові або продажу філіалу, гра буде перевіряти, чи немає в групі клітинок, де кількість філіалів менша (якщо ми будуємо), або більша (якщо продаємо, і якщо виконати ту чи іншу дію неможливо, то відповідна кнопка буде неклікабельна.

Коли функціональна частина механіки готова, можна програмувати візуальну частину, в першу чергу створюються ті самі кнопки побудови, продажу та закриття меню, визначається їх колір, текстовий супровід, а також програмується анімація при наведенні миші. Дуже важливо встановити координати малювання кнопки точно такими, як і координати їх хітбоксу (маски зіткнення з якою взаємодіє миша). Наостанок, буде доречно додати відображення в інтерфейсі того, який рівень має та чи інша клітинка в цей момент гри. Для цього в каталог ресурсів потрібно додати спрайт який символізує рівень філіалу, наприклад це може бути зірка. Після цього залишається прописати в коді материнської клітинки функцію відображення цього спрайту на клітинці у кількості що відповідає рівню філіалу, таким чином всі гравці одразу розуміють наскільки цінним є це поле.

Таким чином, ми маємо готову систему покупки, розбудови та продажу філіалів, що є важливим кроком на шляху до реалізації проекту.

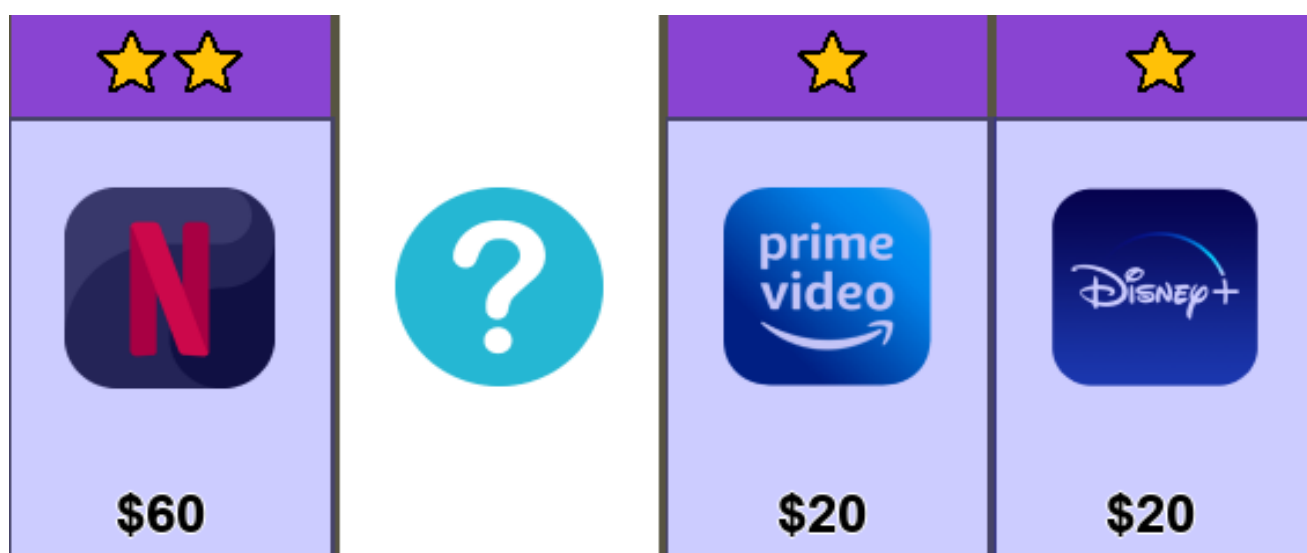


Рис. 2.12. Відображення філіалу у синього гравця

Проте на ігровому полі є не лише клітинки власності, хоча вони, безперечно, є найважливішим елементом ігрового процесу, крім них дошка містить й інші поля з досить цікавими властивостями.

Перша клітинка з унікальними властивостями – це, звісно, клітинка “Старт”. Вона є особливою не лише тому, що саме з неї починається гра, а й тому, що в будь-якій монополії, за правилами, при проходженні кола гравець має отримувати за це винагороду. Реалізується ця функція дуже просто, у коді кожної з фішок потрібно запрограмувати нарахування гравцю певної суми коли фішка опиняється на клітинці “Старт”, навіть якщо вона не зупинилась на ній, а просто проходила повз.

Друга клітинка з цікавими властивостями це клітинка “Шанс”, вона являє собою генератор випадкових подій які можуть як зіграти на руку, так і зашкодити гравцю. На дошці ці клітинки трапляються доволі часто, їх налічується шість одиниць. Щоб реалізувати функціональну частину цих клітинок, доцільно буде створити спеціальний скрипт – “scr_execute_chance” (скрипт виконання шансу). У цьому скрипті прописуються всі можливі події при потраплянні на відповідне поле. В нашому випадку може статися одна з десяти подій, з яких п’ять є позитивними, а інші п’ять – негативні, далі гра просто обирає одну з них випадковим чином і реалізує її. Наостанок залишається, по аналогії з власністю, прописати в події “Створення” контролера ідентифікаційні номери всіх клітинок шанс, щоб програма розуміла, коли цей скрипт потрібно запускати.

Ще однією з клітинок на полі є “Податок”, вона також є досить типовою для всіх варіацій монополії. Її суть дуже проста, при потраплянні на неї гравець має сплатити певну суму. Для їх реалізації потрібно лише прописати в коді створення контролера номери цих клітинок та розмір податку, який стягується з гравця.

Дуже важливою для ігрового процесу є клітинка “В’язниця”, її завдання полягає в тому, щоб затримати гравця якому не пощастить на неї потрапити, на один хід, або, якщо він не бажає чекати, змусити його витратити гроші на викуп своєї фішки з в’язниці. Для того щоб реалізувати цю систему, створюються дві окремі клітинки – “Йдіть до в’язниці” та “В’язниця”. Поле “Йдіть до в’язниці” є лише тригером, який відправляє гравця в саму в’язницю. Клітинка “В’язниця”, своєю чергою, також поділена на дві частини, одна – це просто прохід, через який зазвичай йдуть гравці, а

інша – безпосередньо в'язниця. Щоб реалізувати цю систему, в події “Крок” контролера програмується логіка взаємодії гравця з грою, якщо він опиняється у в'язниці, то перед ним з'являються дві кнопки, при натисканні на кнопку “Pay 200\$” він сплачує гроші та має можливість здійснити хід, а при натисканні “Pass” – залишається у в'язниці до наступного ходу. Також у події “Малювання” потрібно розробити візуальну частину цих кнопок, присвоївши їм колір, розмір, координати та текстове наповнення.

Останньою та найцікавішою клітинкою на дошці є “Казино”, гравець що опиниться на ній, матиме можливість випробувати свою удачу та, потенційно, змінити хід гри на свою користь. Запрограмувати це не складно, потрібно лише створити кнопки які будуть з'являтися перед гравцем при потраплянні в казино і за допомогою яких він зможе зіграти або відмовитись від гри. Відображення кнопок буде відбуватись у два етапи, спочатку гравець приймає рішення, чи хоче він взагалі грати, тут потрібно лише дві кнопки – “Play” та “Pass”, далі, якщо вирішено грати – кнопки якими він обирає розмір своєї ставки, також можна знову додати “Pass”, якщо гравець передумав. Коли гравець здійснює гру в казино, то сума його ставки або подвоюється та повертається на рахунок, або просто згорає. Так само як і для “Шансу”, у контролері прописується функціонал кнопок, а також їх координати та зовнішній вигляд.

Таким чином, ми маємо готовий функціонал для всіх клітинок які мають місце на ігровій дошці, розроблено систему покупки та побудови філіалів, додано різноманітні поля з цікавими подіями.

Наступним кроком в розробці монополії є додавання можливості вибування з гри для тих, хто, навіть продавши всю власність, не має можливості сплатити борги. Оскільки наразі в грі такої системи немає, то в такій ситуації гра просто застрягне і не зможе рухатись далі.

Реалізація цієї механіки відбувається наступним чином: потрібно додати код який, при натисканні лівою клавішею миші на іконку гравця, буде видавати кнопку “Forefeit”, тобто “здатися”, при натисканні на яку гравця буде вилучено з гри, а вся його власність стане знову нейтральною. Щоб запрограмувати цю систему, для кожного об'єкту гравця потрібно додати код, який викликатиме кнопку “здатися” при

натисканні, далі в коді контролера програмується зміна статусу гравця на неактивний, щоб на наступному колі хід до нього вже не переходив, а також вилучення всієї його власності. У події малювання контролера створюється візуал кнопки “здатися”, а в об’єктів гравців додається візуальне відображення його програшу. Таким чином система вибування гравців є готовою, якщо один з них не має можливості сплатити свій борг навіть продавши всю власність, цей гравець натискає на свою іконку, потім на кнопку “здатися”, таким чином він покидає гру.

З плином гри, рано чи пізно, виникне ситуація, коли всі крім одного гравця вже вибули, таким чином цей гравець має стати переможцем. Відповідно, наступним кроком розробки є реалізація механіки перемоги одного з гравців. В об’єкт контролера необхідно додати код, який буде постійно перевіряти, чи немає в грі переможця, тобто чи не залишився лише один гравець, і якщо це так, то викликається меню яке буде відображати цю подію та переносити гравців назад у головне меню.

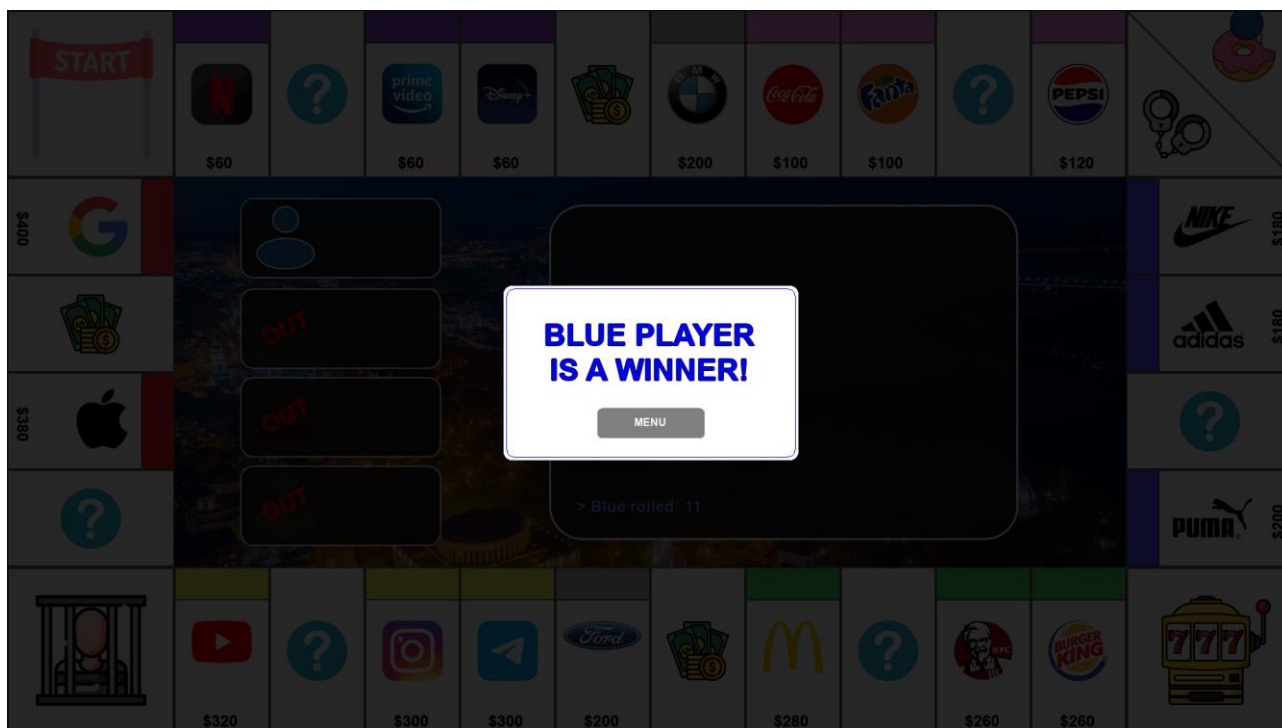


Рис. 2.13. Відображення перемоги гравця

Таким чином ми вже маємо повністю готовий цикл який утворює ігровий процес. Щойно гру запущено, користувач потрапляє у меню, де обирає режим гри та кількість гравців. Далі гра розпочинається, гравці можуть купляти, продавати та розбудовувати філіали, тоді як інші поля, наприклад, шанс, в’язниця і казино

урізноманітнюють ігровий процес. Коли всі гравці крім одного програють, гра завершується, останній гравець що залишився стає переможцем і користувача знову переносить у меню.

Проте не потрібно навіть проводити тестування, щоб зрозуміти, ймовірність того, що всі клітинки однієї групи опиняться в одних руках є низькою, тому в реальних умовах жоден з гравців не зможе будувати філіали й гра буде йти занадто довго. З метою розв'язання цієї проблеми та підвищення динамічності гри, потрібно додати систему обміну клітинками власності між гравцями.

Для того щоб втілити в реальність цю механіку, потрібно, в першу чергу, створити спеціальний скрипт – `scr_execute_trade` (скрипт виконати обмін). У цьому скрипті програмується функціональна основа системи передачі власності між гравцями. Далі потрібно вирішити, як саме буде ініціюватися обмін, наприклад це можна зробити натиснувши на клітинку іншого гравця. Щоб це працювало, у коді кожного об'єкту гравців потрібно додати викликання меню обміну, коли на нього клікає інший гравець у свій хід. Наостанок потрібно створити саме меню здійснення обміну, це відбувається в коді “малювання” об'єкта контролера.

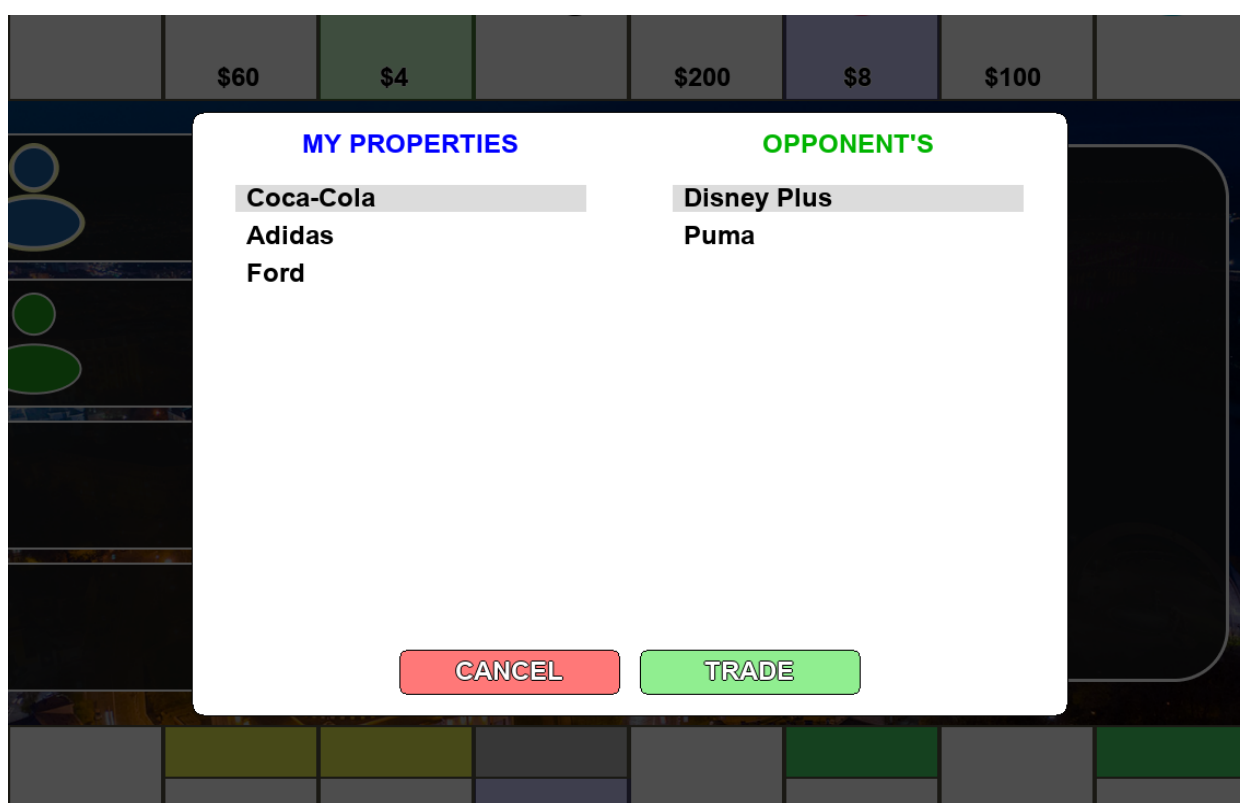


Рис. 2.14. Меню обміну між гравцями

Меню обміну являє собою спеціальне вікно, яке з'являється при натисканні на іншого гравця, у меню міститься список клітинок якими вони володіють. Для того щоб здійснити обмін, кожна сторона повинна вибрати всі поля власності які передбачається передати, для зручності сприйняття, всі обрані клітинки виділяються сірою обводкою. Коли гравці обрали всі клітинки власності які хочуть обмінювати, залишається лише натиснути на кнопку “Trade” і обмін буде здійснено. Якщо з будь-яких причин гравці передумали здійснювати обмін, варто натиснути на кнопку “Cancel”, після цього вікно обміну зникає і гра продовжується.

Реалізація системи обміну власністю була останнім етапом процесу проектування основних ігрових механік, тепер локальний режим гри є цілком грабельним, хоча досі, ймовірно, потребує покращення та доопрацювань.

Підбиваючи проміжні підсумки, на цей момент вже створено завершений алгоритм монополії, починаючи від моменту запуску гри та закінчуючи її завершенням перемогою одного з гравців. Було розроблено ігрову дошку, додано всі необхідні елементи користувацького інтерфейсу, запрограмовано властивості та поведінку внутрішньоігрових об'єктів і реалізовано систему взаємодії між гравцями.

2.3. Інтеграція ШІ, звуку та фінальні доопрацювання

Реалізація режиму гри проти ШІ є останньою, та, мабуть, найскладнішою частиною розробки монополії. Цей етап передбачає створення алгоритму дій ботів який буде, наскільки це можливо, імітувати поведінку реального гравця. Якщо у меню гравець при виборі режиму гри обирає “Гра проти ШІ”, то всі гравці крім синього будуть керуватись ботами та діяти згідно вищезазначеному алгоритму.

Перед початком створення цього алгоритму потрібно чітко визначити, як саме ШІ має діяти у тих чи інших ситуаціях, і вже відштовхуючись від цього програмувати алгоритм. В першу чергу варто встановити спеціальний таймер приблизно на 1 секунду, перед будь-якою дією бота він буде починати відлік і лише по його закінченню дія відбувається, це потрібно щоб ходи ШІ не відбувались моментально і гравець міг відстежувати, що відбувається на полі.

Тепер можна переходити до програмування дій ШІ в тих чи інших ігрових ситуаціях. На початку свого ходу бот здійснює кидок кубиків, так само як і гравець, і

пересувається на відповідну відстань. При потраплянні на клітинку власності яка нікому не належить, бот викупить її, але при умові, що покупка не коштуватиме йому більше як половини його наявного балансу, інакше він відмовиться. При потраплянні на клітинку власності якою володіє інший гравець, а також на клітинки податку, бот сплачує відповідну суму зі свого рахунку. При потраплянні у в'язницю, бот з ймовірністю 50% або викупить себе, або залишиться в ній на один хід, за винятком ситуацій коли на його рахунку менше ніж 400\$, в такому випадку він завжди буде пропускати хід [20]. При потраплянні в казино бот також буде діяти згідно своєї фінансової ситуації, і чим більше в нього грошей, тим на більшу суму він буде грати. Якщо в результаті свого ходу бот не може оплатити свої борги, він почне продавати свої філіали, а якщо цього недостатньо, і клітинки власності, при цьому дотримуючись правил рівномірного продажу. Якщо у бота все добре з фінансами і він вже має всі клітинки власності з однієї групи, на початку свого ходу він буде будувати філіал на випадковій з них, за умови якщо це не коштуватиме йому більше ніж половина його грошей, і знову ж таки, згідно з правилами, рівномірно і не більше одного філіала за хід. Що стосується обміну, якщо запропонувати обмін боту, то він буде оцінювати цінність клітинок якими пропонується обмінятись, якщо він не вважатиме обмін вигідним, то кнопка буде неклікабельною, цінність клітинки бот оцінюватиме по актуальній вартості оренди на ній. Наостанок, якщо бот навіть продавши все, що у нього є, не може сплатити свої борги, він вибуває з гри.

Цей алгоритм насправді є доволі простим, він не передбачає складних розрахунків з сотнями або тисячами можливих ходів наперед, як це відбувається у високобюджетних проєктах ігор типу “Монополія”, але це саме те, що цілком можливо реалізувати з наявними ресурсами, при цьому зберігаючи відчуття гри проти справжнього ІІІ.

Функціональна частина цієї системи здебільшого програмується в події “крок” об'єкта глобального контролера, там визначається поведінка бота при потраплянні на ті чи інші клітинки. Також створюються спеціальні скрипти – `scr_bot_logic_build`, `scr_bot_sell_assets` і `scr_get_cell_value`, завданням яких є програмування побудови, продажу та обміну філіалами відповідно.

З погляду візуальної частини, на цьому етапі потрібно лише внести невеликі зміни до коду меню обміну, щоб коректно відобразити неклікабельну кнопку обміну. Після реалізації всіх цих кроків, ми нарешті маємо повністю готовий режим гри проти ІІІ.

Нарешті, останнім етапом створення монополії є додавання звуків у гру. Звукове супроводження є важливим, оскільки воно суттєво впливає на відчуття гри користувачем та створення бажаної атмосфери [25].

Додавання звуку є відносно простим завданням, спершу, з цією метою потрібно у менеджері ресурсів створити файли для всіх звуків, що матимуть місце в грі, в нашому випадку це – звук кидка кубиків, звук грошового трансферу, звук перемоги гравця і загальна фоновіа мелодія. Далі потрібно створити або завантажити з відкритих джерел відповідні звуки, наприклад, можна скористатись онлайн-сервісом “Freesound” [18]. З цього сервісу звуки завантажуються на пристрій, за потреби редагуються та обрізаються, а згодом інтегруються в середовище GameMaker. Тепер залишається прописати в коді відтворення цих звуків у певних ігрових ситуаціях. Звук кидка кубиків, наприклад, прив’язується до функції здійснення ходу гравцем, звук перемоги – до моменту коли впливає меню перемоги одного з гравців, а звук грошей, відповідно, до будь-якої зміни балансу гравців. Фонова мелодія, своєю чергою, просто починає грати щойно гру запущено і до самого моменту виходу з гри.

Нарешті, після завершення інтеграції ІІІ та звуку в гру, процес розробки можна вважати майже завершеним. Останнім, але однак, дуже важливим є процес тестування отриманого продукту та доопрацювань. Оскільки в результаті розробки ми отримали тисячі рядків програмного коду, не важко уявити, що десь могла бути допущена помилка, яку не вдалось помітити одразу, саме тому, на цьому етапі потрібно тестувати якомога більше можливих ігрових сценаріїв і перевіряти, чи правильно працює програма. При виявленні багів їх слід обережно виправляти, при цьому не зламавши іншу частину програми. Після проведення ретельної перевірки, тестування та виправлення всіх недоліків гри, процес розробки монополії є завершеним.

РОЗДІЛ 3. АНАЛІЗ РЕЗУЛЬТАТУ РОЗРОБКИ, ПУБЛІКАЦІЯ ТА ПЕРСПЕКТИВИ РОЗВИТКУ

3.1. Аналіз та оцінка результату роботи

Аналіз результату розробки монополії дозволяє оцінити відповідність кінцевого програмного продукту поставленим на початку проекту цілям та завданням. Загалом, всю основну логіку функціонування гри та взаємодії її з користувачем було реалізовано в повному обсязі. Після запуску гри користувач потрапляє у головне меню, де він спочатку обирає режим гри, а потім кількість гравців, програма запам'ятовує його вибір та далі діє відповідним чином. Далі гравець опиняється на ігровому полі, де вступає в дію механіка руху фішок, що базується на генерації випадкових чисел у діапазоні від 2 до 12, що імітує кидок двох шестигранних кубиків. Кожна фішка переміщується по клітинках вперед на ту кількість кроків, яка випала на кубуку. Весь ігровий процес супроводжується роботою внутрішньої консолі, яка у реальному часі виводить текстові повідомлення про кожну подію, забезпечуючи прозорість ігрової логіки для користувача.

Також вдалося реалізувати систему взаємодії гравця з клітинками власності, а саме: покупку власності, побудову філіалу та продаж. Ця система працює так само, як у всіх іграх типу монополії, спочатку гравець має заволодіти всіма клітинками з однієї групи, і тільки потім він може будувати там філіали, тим самим збільшуючи орендну плату. При цьому, в коді прописано декілька умов, що регулюють цей процес, по-перше, купувати чи продавати філіали можна тільки рівномірно, по-друге, можна купити лише один філіал за хід, однак продавати скільки завгодно.

Окрім клітинок власності на полі присутні сектори спеціальних подій, які роблять ігровий процес більш цікавим і непередбачуваним. Клітинка “Шанс” є генератором випадкових подій які можуть як нашкодити, так і допомогти гравцю, клітинка “Казино” пропонує йому можливість ризикнути своїми грошима заради потенційного виграшу, а “В’язниця” примусово затримує гравця, результаті чого він повинен, або пропустити хід, або платити викуп.

Для підвищення динамічності гри було створено систему обміну, вона дозволяє гравцям обмінюватись своєю власністю, тим самим відкриваючи простір для різних

Перед створенням кінцевого файлу важливо виставити йому відповідні налаштування. По-перше, варто обрати іконку для гри, створити її можна так само, як і всі інші елементи ігрового інтерфейсу – за допомогою сервісу Figma, інакше в якості іконки гри буде відображатись логотип самого GameMaker. Також на цьому етапі потрібно провести оптимізацію ресурсів, зокрема згрупувати графічні атласи для зменшення навантаження на відеопам'ять та вилучити всі дебаг-функції, які використовувалися для налагодження роботи гри. Це дозволить отримати чистий програмний код, готовий до експлуатації кінцевим користувачем.

Для створення кінцевого файлу було використано вбудований компілятор Virtual Machine, він перетворює вихідний код GML у байт-код, зрозумілий для ігрового рушія. Таким чином було сформовано набір файлів, що складають структуру ігрового пакета, основою якої є виконуваний файл із розширенням .exe, цей компонент гри відповідає за ініціалізацію запуску рушія та обробку всіх команд користувача. Також важливу роль відіграє ресурсний файл data.win – найважливіший архівований елемент, у якому містяться всі ігрові активи, включаючи спрайти, об'єкти, скрипти та системну логіку. Відокремлення ресурсів у цей файл забезпечує цілісність архітектури та надійно захищає графічний контент від випадкових змін.

Завершальним кроком публікації гри стало пакування всіх сформованих файлів у єдиний ZIP-архів. Такий підхід дозволяє користувачу просто розпакувати архів у будь-яку папку на жорсткому диску та запустити виконуваний файл для початку гри, не вдаючись до складних процедур інсталяції. Відсутність необхідності в установці робить продукт зручним для швидкого поширення та тестування на різних пристроях. Таким чином, розроблений програмний продукт є повністю автономним і готовим до запуску користувачами.

3.3. Аналіз можливостей вдосконалення та перспектив розвитку гри

Як і у будь-якого програмного продукту, процес вдосконалення гри ніколи не є завершеним, а можливості розвитку ніколи не є вичерпаними. Навіть після багатьох тестувань, коли гра вже була опублікована, в ній з плином часу можуть виявлятися ті чи інші баги, це означає, що розробник має регулярно випускати оновлення з метою їх виправлення.

Втім окрім полагодження багів, розробник також має подбати про покращення та урізноманітнення ігрового процесу, це є необхідною частиною життєвого циклу будь-якої гри. Саме тому, на основі проведеного раніше аналізу нашого продукту, важливо окреслити можливі напрями розвитку монополії.

Найбільш багатообіцяльною є можливість додавання в гру онлайн-режиму гри. Реалізація повноцінного онлайн-режиму дозволить проєкту вийти за межі використання лише на одному локальному пристрої та значно збільшить обсяг аудиторії, адже у світі є чимало гравців, які хотіли б пограти зі своїми друзями, або навіть з випадковими людьми, незалежно від їх географічного розташування. Після додавання онлайн режиму гри логічно слідує додавання системи рейтингу гравців, внутрішньоігрових досягнень, та, можливо, навіть турнірів. Реалізація цього задуму вимагатиме переходу від локальної архітектури до серверної моделі, яка дозволить синхронізувати стан ігрового поля, фінансові показники гравців та статус власності в реальному часі.

Також важливим напрямком розвитку гри є вдосконалення алгоритмів штучного інтелекту. Поточна версія ШІ досить вдало імітує базову поведінку гравця, але, однак, існує чималий простір для поглиблення аналізу та оптимізації системи прийняття рішень. Наприклад, можна зробити так, щоб коли боти приймають рішення про покупку клітинки власності, вони не лише дивилися на свій баланс, а ще й оцінювали потенційну цінність цієї клітинки для себе, або для інших гравців, враховуючи ситуацію на дошці. Таким чином можна додати можливість для користувача обирати складність ботів, проти яких він хоче зіграти.

Звісно, існує великий простір для покращення та оптимізації гри також з технічної та естетичної думки. Зокрема, використання можливостей рушія GameMaker для створення складних частинок, динамічного освітлення та плавної анімації інтерфейсу дозволить наблизити якість продукту до комерційних аналогів. Окрім базових спрайтів, подальша модернізація передбачає впровадження шейдерів для створення візуальних ефектів або підсвічування активних зон, що значно покращить сприйняття ігрового простору та допоможе гравцям краще орієнтуватися в ігрових подіях. У поєднанні з адаптивним звуковим супроводом, що реагує на

критичні моменти гри (наприклад, стан банкрутства чи успішні транзакції), це забезпечить глибоке занурення користувача в атмосферу гри.

Тепер, коли ми маємо розуміння основних напрямків розвитку самої монополії, необхідно також подумати про процес публікації та дистрибуції продукту, адже це відкриє значний потенціал для його масштабування та виходу на професійний рівень завдяки архітектурній гнучкості рушія GameMaker. Пріоритетним кроком у цьому напрямі є розміщення проєкту на спеціалізованих майданчиках, таких як itch.io [26], що дозволить забезпечити зворотний зв'язок із першими користувачами через систему відгуків і коментарів. Подальше масштабування передбачає вихід на комерційні платформи цифрової дистрибуції, зокрема Steam, що потребуватиме інтеграції API Steamworks для реалізації таких функцій, як глобальна таблиця лідерів, хмарні збереження прогресу та система ігрових досягнень. Важливим аспектом дистрибуції є реалізація кросплатформного потенціалу проєкту: використання додаткових модулів експорту дозволяє портувати гру на мобільні операційні системи Android та iOS, а також адаптувати її під формат HTML5 для запуску безпосередньо у веббраузерах. Такий підхід значно знижує поріг входу для гравців і дозволить сформувати стабільну екосистему навколо гри та забезпечити продукту тривалий життєвий цикл на ринку відеоігор.

Підсумовуючи, можна стверджувати, що перспективи розвитку монополії дозволять перетворити локальний проєкт на повноцінний продукт, готовий до виходу на широкий ринок. Реалізація цих планів допоможе не лише покращити технічну сторону гри, а й значно розширити аудиторію завдяки новим платформам та мережевим можливостям. Це підтверджує, що розробка має серйозний потенціал для зростання, а обрані інструменти дають змогу гнучко розвивати гру в майбутньому.

ВИСНОВОК

В результаті виконання кваліфікаційної роботи було реалізовано повний цикл розробки цифрової версії настільної гри “Монополія”. Цей процес охопив всі етапи створення відеогри: від фундаментального аналізу предметної області та проєктування складних ігрових механік до безпосередньої технічної реалізації, ітераційного тестування та фінальної підготовки дистрибутиву. Під час розробки було не лише відтворено класичні правила настільної гри, а й адаптовано їх до сучасного цифрового середовища.

Перед початком розробки було проведено детальний огляд начвної літератури на відповідну тему, а також аналіз ігрових аналогів у жанрі економічної стратегії. Дослідження літератури в галузі геймдизайну, економічного моделювання та психології користувацького досвіду дозволило сформувати чітку теоретичну базу для побудови збалансованої ігрової логіки. Паралельно з цим, детальний порівняльний аналіз популярних цифрових версій «Монополії» допоміг виявити як ключові переваги успішних проєктів, так і їх певні недоліки. На основі отриманих знань було створено покроковий план розробки монополії який став дорожньою картою для втілення проєкту в життя.

На початковому етапі особлива була приділена обґрунтуванню вибору інструментарію розробки. Використання ігрового рушія GameMaker у поєднанні з мовою програмування GameMaker Language дозволило ефективно побудувати архітектуру гри. Це рішення забезпечило гнучкість у керуванні ігровими об'єктами, що було критично важливим для створення динамічного ігрового поля з великою кількістю взаємозалежних елементів. Процес розробки підтвердив, що обрана технологія є оптимальною для створення високопродуктивних двовимірних стратегій, оскільки вона надає широкі можливості для оптимізації графічних ресурсів та забезпечує стабільну роботу додатка без надмірних вимог до апаратної частини комп'ютера користувача.

Результатом практичної частини роботи стала успішна реалізація цілісного ігрового циклу, що включає механіки випадкового генерування чисел, переміщення фішок, а також складну систему фінансових розрахунків. Розроблена економічна

модель гри охоплює процеси купівлі клітинок власності, динамічне нарахування оренди залежно від рівня розбудови філіалів, механіки оподаткування та випадкових подій у секторах «Шанс» і «Казино». Також створено алгоритми штучного інтелекту які дозволяють імітувати стратегічну поведінку гравця, включаючи оцінку вигідності придбання нерухомості та ініціювання процесів обміну активами, що робить гру цікавою та конкурентною.

Подальше тестування готового продукту підтвердило працездатність усіх модулів системи. У ході тестування було успішно усунено виявлені помилки в логіці взаємодії об'єктів та забезпечено коректну синхронізацію візуальних ефектів зі звуковим супроводом. Завершальним етапом стала підготовка автономного дистрибутиву, який у форматі містить усі необхідні компоненти для запуску гри на операційній системі Windows без попередньої інсталяції.

Наостанок можна зазначити, що виконання даної кваліфікаційної роботи дозволило не лише створити функціональний ігровий продукт, а й отримати навички володіння принципами алгоритмізації та проектування програмних систем. Було важливо не просто скопіювати правила, а спроектувати систему, яка була б інтуїтивно зрозумілою для користувача та технічно стабільною. Практична реалізація ігрового процесу дозволила глибше зрозуміти принципи побудови архітектури програмного забезпечення. Стало зрозуміло, що навіть ідеально написана програма не буде працювати належним чином без якісного зворотного зв'язку з користувачем, кожна дія на ігровому полі повинна бути зрозумілою та візуально підкріпленою.

Таким чином, розроблена гра стала не лише втіленням програмної логіки в ігровому середовищі, а й наочним результатом трансформації теоретичних знань у завершений, автономний програмний продукт, що повністю відповідає поставленій меті та готовий до практичної експлуатації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гладченко О.В., Пригоровський В.Д. Методологічні аспекти проектування цифрових імітаційних середовищ для оптимізації управлінських рішень в економічних системах. // Економічна кібернетика: сучасні інформаційні технології в управлінні : Всеукр. наук.-практ. інтернет-конф., 3-4 берез. 2026 р. : тези доп. – Дніпро: Український державний університет науки і технологій. – С. 196-201. URL : https://nmetau.edu.ua/file/zbirnik_naukovih_pratst_2026.pdf (дата звернення: 11.03.2026).
2. Самохвалов О. Ю., Пасічник В. В. Проектування інтерфейсів користувача: навчальний посібник. Львів: Новий Світ-2000, 2023. 210 с.
3. Сиротюк В. О., Зайцева Т. В. Методи та системи штучного інтелекту: підручник. Київ: Академвидав, 2022. 340 с.
4. Ткачук В. В. Цифрові технології в освітньому процесі: гейміфікація та інтерактивні середовища: монографія. Кривий Ріг: КДПУ, 2023. 244 с.
5. Трофименко О. Г., Прокоп Ю. В., Швайко І. Г. Створення програмних продуктів: навч. посібник. Одеса: Фенікс, 2021. 248 с.
6. Шеховцов В. А., Коренєва Л. В. Архітектура програмного забезпечення та проектування складних систем: навчальний посібник. Харків: ХНУРЕ, 2022. 188 с.
7. Adobe Photoshop. Made for your pro era. URL: <https://www.adobe.com/products/photoshop.html> (дата звернення: 03.03.2026).
8. Based. Основи ігрового дизайну : практичний посібник. URL: <https://www.based.ua/uk/osnovy-igrovogo-dyzajnu-praktychnyj-posibnyk/> (дата звернення: 06.03.2026).
9. Bateman C. Game Writing: Narrative Skills for Videogames: monograph. London: Bloomsbury Academic, 2021. 256 p.
10. Belovolchenko H. Як працювати з ігровими механіками й чому не всі з них вдалі. URL: <https://gamedev.dou.ua/articles/how-to-work-with-game-mechanics/> (дата звернення: 25.02.2026).

11. Ben A. The Maths of Monopoly. URL: <https://www.marlbroughcollege.org/wp-content/uploads/2024/05/Ben-A-the-Maths-of-Monopoly.pdf> (дата звернення: 20.03.2026).
12. Bewersdorff J. Luck, Logic, and White Lies: The Mathematics of Games: monograph. Boca Raton: CRC Press, 2021. 556 p.
13. Chan M. Handmade Designs: The New Trust Signal. URL: <https://www.nngroup.com/articles/handmade-designs/> (дата звернення: 09.03.2026).
14. Durango B. Monopoly Probabilities. URL: <https://www.durangobill.com/Monopoly.html> (дата звернення: 10.03.2026).
15. Fessenden T. Three Pillars of User Delight. URL: <https://www.nngroup.com/articles/pillars-user-delight/> (дата звернення: 28.02.2026).
16. Figma. Figma Design Service. URL: <https://www.figma.com/> (дата звернення: 02.02.2026).
17. Flaticon. Flaticon Vector Icon Database. URL: <https://www.flaticon.com/> (дата звернення: 11.02.2026).
18. Freesound. Freesound Audio Database. URL: <https://freesound.org/> (дата звернення: 16.03.2026).
19. Game Developer. 6 Game UI You Should Study. URL: <https://www.gamedeveloper.com/design/6-game-ui-you-should-study> (дата звернення: 22.03.2026).
20. GameMaker. Start Making Games. URL: <https://gamemaker.io/> (дата звернення: 13.02.2026).
21. GeeksforGeeks. Minimax Algorithm in Game Theory. URL: <https://www.geeksforgeeks.org/dsa/minimax-algorithm-in-game-theory-set-1-introduction/> (дата звернення: 21.03.2026).
22. Gibbons S., Wang H. Design Process Isn't Dead, It's Compressed. URL: <https://www.nngroup.com/articles/design-process-isnt-dead/> (дата звернення: 18.03.2026).
23. Godot. Your free, open-source game engine. URL: <https://godotengine.org/> (дата звернення: 15.02.2026).

24. Gordon K. Using Color to Enhance Your Design. URL: <https://www.nngroup.com/articles/color-enhance-design/> (дата звернення: 27.03.2026).
25. Hodent C. The Gamer's Brain: How Neuroscience and UX Can Impact Video Game Design: monograph. Boca Raton: CRC Press, 2020. 270 p.
26. itch.io. itch.io Platform. URL: <https://itch.io/> (дата звернення: 25.03.2026).
27. Johnson J. Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines: monograph. San Diego: Morgan Kaufmann, 2020. 272 p.
28. Krause R. Closing the Loop: What to Do After a Design Critique Ends. URL: <https://www.nngroup.com/articles/after-design-critique/> (дата звернення: 02.04.2026).
29. Magnific. The creative platform to direct your best work. URL: <https://www.magnific.com/> (дата звернення: 17.02.2026).
30. Moran K. GenUI vs. Vibe Coding: Who's Designing? URL: <https://www.nngroup.com/articles/genui-vs-vibe/> (дата звернення: 15.03.2026).
31. Nilsson A. Exploring strategies in Monopoly using Markov chains and simulation. URL: <https://uu.diva-portal.org/smash/get/diva2:1471765/FULLTEXT01.pdf> (дата звернення: 17.03.2026).
32. Samsonov P. The Product Triad: Design's Role. URL: <https://www.nngroup.com/articles/the-product-triad-designs-role/> (дата звернення: 10.03.2026).
33. Schwab B. AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors: monograph. Boca Raton: CRC Press, 2022. 310 p.
34. The Noun Project. Free Icons and Stock Photos, Made by Humans. URL: <https://thenounproject.com/> (дата звернення: 14.02.2026).
35. Thorn A. Introduction to Game Design and Development: textbook. Boston: Charles River Media, 2024. 415 p.
36. Tsalikova D. Невидима рука ринку в ігровій економіці. URL: <https://gamedev.dou.ua/blogs/decentralized-game-design/> (дата звернення: 05.03.2026).
37. Unity. Build Great. URL: <https://unity.com/> (дата звернення: 15.02.2026).
38. Unreal Engine. We make the engine. You make it Unreal. URL: <https://www.unrealengine.com/> (дата звернення: 15.02.2026).

39. Wang. H. Small by Design: The Strength of Lean Design-System Teams. URL: <https://www.nngroup.com/articles/lean-design-system-teams/> (дата звернення: 01.03.2026).

40. Wolfram MathWorld. Dice. URL: <https://mathworld.wolfram.com/Dice.html> (дата звернення: 10.02.2026).