УДК 004.415.2
DOI https://doi.org/10.32782/2663-5941/2025.3.2/34

Майданюк В.П.
Вінницький національний технічний університет

Складанюк О.О.
Вінницький національний технічний університет

ГРАФІЧНИЙ РЕДАКТОР ДЛЯ РЕДАГУВАННЯ ДЕСКТОПНИХ ВІДЕОІГОР

У статті виконано аналітичний огляд методів та засобів для редагування сцен у 2D-відеоіграх. Обґрунтовано актуальність створення подібного інструменту з урахуванням поширеності інді-розробок і потреби в інтуїтивно зрозумілих редакторах. Проаналізовано технічні аспекти реалізації функціоналу, таких як інтерактивне переміщення об'єктів (drag-and-drop), редагування властивостей об'єктів сцени у реальному часі, управління шарами та компонентами сцени. Розглянуто підходи до збереження структури сцени у форматі JSON, що дозволяє легко експортувати та імпортувати дані про ігрові рівні. Особливу увагу приділено використанню 2D-графічної бібліотеки Pixi.js як рушія для рендерингу сцени, а також принципам побудови архітектури на основі об'єктно-орієнтованого підходу. Здійснено аналіз використання подієво-орієнтованої моделі для забезпечення взаємодії між компонентами редактора. Окремо проаналізовано перспективи розширення функціоналу редактора, зокрема реалізацію сітки для вирівнювання об'єктів, систему undo/redo, підтримку кількох сцен, та інструменти для створення складніших ігрових механік. Визначено напрями подальших досліджень, зокрема створення адаптивної системи плагінів для розширення функцій редактора. Основним науково-практичним результатом дослідження є визначення ключових компонентів, потрібних для побудови ефективного редактора 2D-сцен, а також реалізація базової версії редактора, що може бути використана як основа для подальшого розвитку. Основними перевагами запропонованого редактору є модульність інтерфейсу, легка масштабованість проекту, простота розробки та підтримки, збереження сцен у JSON, наявність системи шарів, drag-and-drop редагування, базова підтримку undo/redo, інтеграція з Pixi.js як візуальним рушієм, платформнонезалежність. Практична цінність роботи полягає у можливості використання розробленого інструменту для побудови прототипів та ігрових рівнів без необхідності написання коду, що сприяє прискоренню процесу розробки 2D-ігор, дозволяючи розробникам зосередитися на ігровій логіці та дизайні, а не на технічних деталях реалізації сцени. Запропонований редактор може бути інтегрований у навчальні програми для ознайомлення студентів із принципами побудови ігрових середовищ та графічного програмування.

Ключові слова: *графічний редактор, відеоігри, сцени, 2D, drag-and-drop, Pixi.js, JSON, undo/redo, шари, об'єктно-орієнтоване програмування.*

Постановка проблеми. У сучасну епоху ігрової індустрії все більше людей отримують доступ до інструментів створення власних відеоігор. Зокрема, зростає популярність так званих інді-розробок (англ. *indie*, скорочено від *independent* – створення ігор командами або окремими розробниками, що не залежать від великих видавців), які потребують простих, ефективних та інтуїтивно зрозумілих інструментів для створення та редагування ігрових сцен. Багато з таких проєктів мають обмежені ресурси, тому виникає потреба у графічних редакторах, які дозволяють створювати та змінювати 2D-сцени без потреби у глибоких знаннях програмування. Одним із перспективних рішень є створення універсального редактора, що підтримує візуальне редагування, організацію об'єктів сцени та збереження її структури у зручному форматі, придатному для імпорту в ігровий рушій [1].

Аналіз останніх досліджень і публікацій. У статті здійснено аналітичний огляд сучасних інструментів для редагування сцен у 2D-відеоіграх. Аналізуються такі продукти:

- Unity і Unreal Engine – як професійні рушії, що мають потужні, але складні в освоєнні інструменти для редагування сцен;
- Tiled Map Editor – потужний редактор, але з ускладненням при інтеграції кастомних об'єктів;
- LDtk – редактор з автоматизованими тайлами, проте орієнтований здебільшого на платформери;

• Ogmo Editor – легкий і зручний для новачків, однак має обмежену гнучкість і функціональність.

Проблеми, що залишаються невирішеними:

• Відсутність універсального інструменту, що поєднує простоту, модульність, масштабованість, drag-and-drop редагування, збереження у JSON та підтримку undo/redo.

• Недостатня підтримка кастомізації, анімацій, GUI-подій та плагінів у більшості наявних редакторів.

Огляд і аналіз аналогів. Серед популярних інструментів для створення 2D-ігор слід відзначити Tiled, LDtk, Ogmo Editor. Tiled підтримує багато типів карт, але має складну інтеграцію кастомних об›єктів. LDtk забезпечує автошари та інтуїтивний інтерфейс, але орієнтований переважно на платформери. Ogmo Editor – простий, зручний, але з обмеженою функціональністю. Таким чином, існує нереалізована ніша – створення редактора з гнучкою архітектурою, системою шарів, підтримкою undo/redo, інтеграцією з рушіями та простотою використання.

Tiled Map Editor – це потужний, безкоштовний та відкритий редактор карт для 2D-ігор, який підтримує різні типи карт: ортогональні, ізометричні та гексагональні. Загальний вигляд інтерфейсу даного редактору наведено на рис. 1. Основні можливості Tiled Map Editor такі:

– підтримка кількох шарів тайлів та об’єктів, що дозволяє створювати досить складні сцени [6];

– можливість додавати прямокутники, еліпси, полігони та інші об’єкти з індивідуальними властивостями, що фактично надає гнучкості об’єктним шарам;

– можливість створення та повторного використання шаблонів для стандартних елементів;

– підтримка експорту у різні формати: TMX (XML), JSON, Lua та інші, що забезпечує сумісність із багатьма ігровими рушіями.

LDtk (Level Designer Toolkit) – це сучасний редактор рівнів, створений автором гри Dead Cells. Він орієнтований на простоту використання та досить високу ефективність [7]. Загальний вигляд інтерфейсу даного редактору наведено на рис. 2.

Основні особливості LDtk такі:

– інтуїтивно зрозумілий, сучасний інтерфейс користувача з акцентом на зручність;

– наявність системи автоматичне розміщення тайлів (auto-tiling) на основі правил, що значно пришвидшує процес створення рівнів;

– генерація PNG-файлів для візуалізації та JSON-файлів для даних, що полегшує інтеграцію з ігровими рушіями.

– підтримка кастомних полів, що забезпечується можливістю додавання власних полів до об’єктів для зберігання додаткової інформації.

Ogmo Editor – це легкий, відкритий редактор рівнів, розроблений для інді-розробників [8]. Загальний вигляд його інтерфейсу наведено на рис. 3. Основними перевагами редактора є:

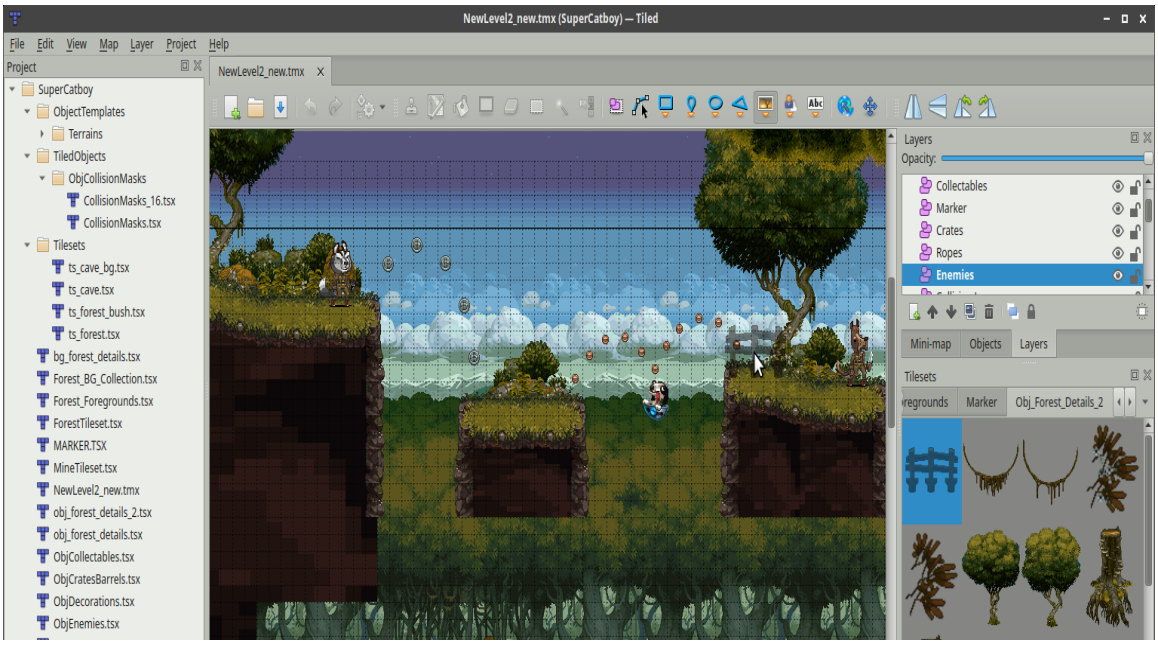

Рис. 1. Інтерфейс редактора Tiled Map Editor

– передбачено проєктно-орієнтований підхід, де всі налаштування зберігаються у проєкті, що забезпечує узгодженість між рівнями;

– підтримка різних типів шарів (тайлових, сіткових, об'єктних та декоративних) для гнучкого дизайну рівнів;

– передбачає зручний формат для інтеграції з багатьма ігровими рушіями (експорт у JSON);

– простота використання завдяки наявності інтуїтивно зрозумілого інтерфейсу, що дозволяє швидко освоїтися навіть новачкам.

Вищенаведені аналоги графічних редакторів для 2D-ігор мають спільний набір базових функцій, таких як підтримка шарів, drag-and-drop інтерфейс, збереження сцен у форматі JSON та система undo/redo. Ці функції створюють зручне середовище для візуального редагування ігрових сцен без потреби у прямому втручанні у код. Завдяки використанню таких бібліотек, як Pixi.js, і сучасних підходів до управління станом (наприклад, Zustand), забезпечується гнучкість і масштабованість редакторів. У поєднанні з можливістю інтеграції з рушіями або власними пайплайнами розробки, ці редактори стають незамінними інструментами, особливо у сфері інді-розробки.

Постановка завдання. Метою статті є розширення функціональних можливостей графічного редактору для 2D-сцен, визначення архітектурних рішень і ключових компонентів для реалізації функціоналу, необхідного для редагування ігрових рівнів, а також побудова базової

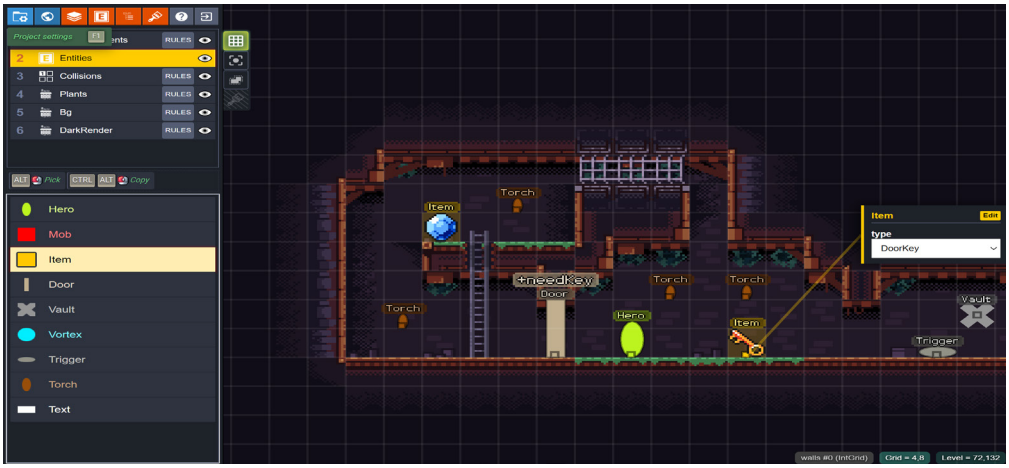

Рис. 2. Інтерфейс редактора LDtk

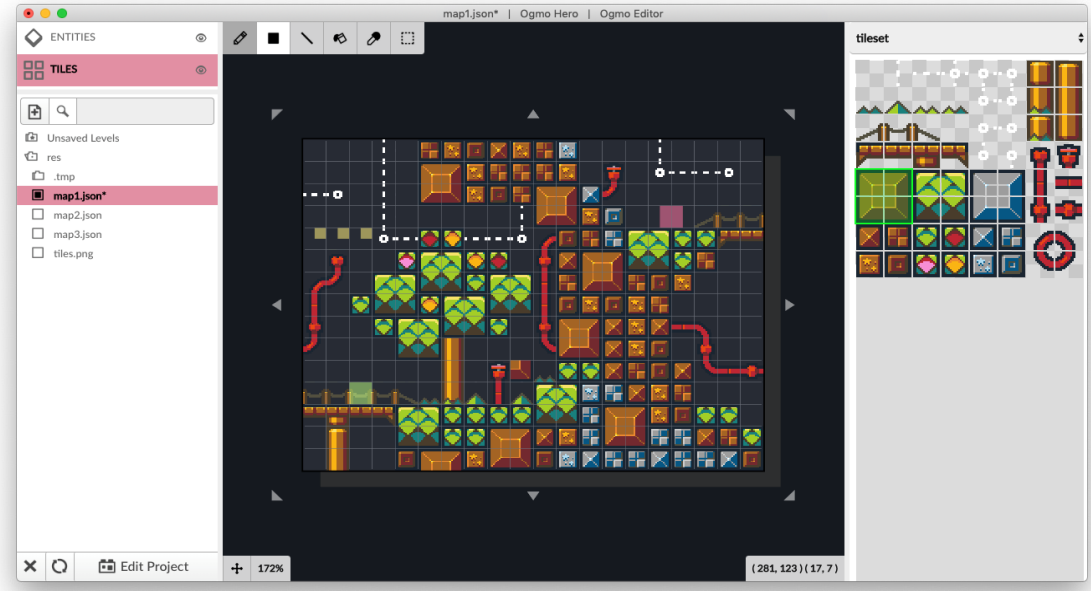

Рис. 3. Інтерфейс редактора Ogmo Editor

версії інструменту, що може бути використана у процесі інді-розробки відеоігор.

Виклад основного матеріалу. У типовому редакторі сцен для 2D-ігор мають місце такі ключові компоненти:

1. Сцена та об'єкти зберігаються у форматі JSON, що дозволяє легко імпортувати/експортувати дані та інтегрувати з рушієм.

2. Сцена підтримує ієрархічну структуру, що дозволяє створювати вкладені об'єкти, групи, шари з різними рівнями видимості.

3. Інтерфейс drag-and-drop забезпечує зручне переміщення, масштабування та обертання об'єктів мишею у редактор [3].

4. Підтримка історії змін – необхідна функціональність, що дозволяє користувачеві повертати попередні стани сцени (Undo/redo).

5. Панель властивостей дозволяє змінювати позицію, розмір, текстуру, поведінку об'єктів тощо.

У розробці редактора часто використовують бібліотеки на кшталт Pixi.js для рендерингу сцени, React або Vue для побудови інтерфейсу користувача (User Interface, UI), Zustand або Redux – для керування станом. У складніших рішеннях можливе використання WebAssembly для пришвидшення обчислень, а також Canvas/ WebGL для рендерингу великої кількості об'єктів у сцені без втрати продуктивності.

На сьогоднішній день багато популярних ігрових рушіїв, таких як Unity та Unreal Engine, пропонують розробникам потужні інструменти для редагування сцен та об'єктів. Ці інструменти дозволяють змінювати візуальні та функціональні аспекти гри без необхідності переписувати код. Unity – це потужний і дуже популярний багатоплатформний рушій для створення комп'ютерних ігор, а також інтерактивних 2D і 3D-додатків. Він надає розробникам усе необхідне для створення ігор. Саме тому його використовують фахівці з усього світу, і багато хто з них вважає Unity номером 1 у світі розробки ігор [4]. Принцип роботи в Unity передбачає ряд особливостей, які розробники мають врахувати під час створення гри. Нижче розглянемо основні з них детальніше.

1. Першою особливістю є створення сцени. Сцена – це простір, де розміщуються та взаємодіють об'єкти гри. Їх може бути декілька на кожному рівні. В Unity можна створювати та редагувати сцени, визначати компоненти оточення, задавати вигляд та рівні освітлення, камери і т. д.

2. Ще однією важливою особливістю є додавання об'єктів. В Unity об'єкти – це основні будівельні блоки гри. Тобто необхідно додавати різні об'єкти до сцени (такі як персонажі, предмети, перешкоди тощо), а також визначити їх взаємодію та поведінку.

3. Ще одна особливість передбачає застосування скриптів та кодування. За допомогою скриптів необхідно створювати поведінку об'єктів та керувати ігровою логікою. За допомогою коду можна створювати унікальні функції, визначити умови перемоги або поразки, обробляти введення користувача та багато іншого. Доцільно зазначити, що вся робота з Unity здійснюється мовою C#.

4. На етапі, коли гра створена, її необхідно виконати її ретельне тестування на різних пристроях та платформах. Тестування допомагає виявити помилки, покращити геймплей та оптимізувати продуктивність, для того, щоб забезпечити ефективне функціонування гри на різних пристроях.

5. На останньому кроці гра публікується на вибраних платформах, таких як комп'ютери, мобільні пристрої, ігрові приставки та інше. Unity надає ряд корисних інструментів для експорту та публікації гри на різних платформах, що дозволяє досягти широкої аудиторії гравців [2].

А тепер розглянемо особливості ігрового рушія Unreal Engine. Unreal Engine – інструмент для створення ігор і сцен із використанням 3D-моделей, розроблений компанією Epic Games. Особливості рушія такі:

1. Він надає засоби для створення дивовижних графічних ефектів: високоякісне освітлення, рендеринг (формування зображення або відео з тривимірної сцени) та підтримка віртуальної реальності.

2. Він дозволяє моделювати об'єкти, відтворювати їхній рух і взаємодію з навколишнім середовищем досить реалістично.

3. Він може бути використаний для розробки ігор на різних платформах: комп'ютерах, консолях, мобільних пристроях і у віртуальній реальності.

Він використовує мову програмування C++, а також має систему розширень та графічний інтерфейс для того, щоб зробити гру, не знаючи кодингу [5].

Створено власний графічний редактор, що має загальний інтерфейс, наведений на рис. 4. Його порівняння з аналогами наведено

у табл. 1. Даний графічний редактор поєднує гнучкість сучасних веб-технологій з можливістю запуску на десктопі завдяки використанню React.js для побудови інтерфейсу та Electron для створення кросплатформного десктопного застосунку. Редактор надає функції створення власних об'єктів з можливістю вставляти їх у обхідну сцену. Основними перевагами даної розробки є:

– використання компонентного підходу React дозволяє легко масштабувати проєкт та підтримувати розширюваність (наприклад, додавання нових панелей, типів об'єктів тощо);

– завдяки вебстеку розробка виконується значно швидше порівняно з C++ або C#, що використовуються в аналогах (Tiled, LDtk);

– сцена формується як структура з об'єктами, що легко експортується у формат JSON для подальшої інтеграції з ігровим рушієм;

– висока гнучкість архітектури, що досягається шляхом реалізації системи шарів, drag-and-drop редагування, базової підтримки undo/redo, інтеграції з Pixi.js як візуальним рушієм;

– платформонезалежність, оскільки Electron дозволяє запускати редактор на Windows, macOS та Linux без необхідності специфічної компіляції.

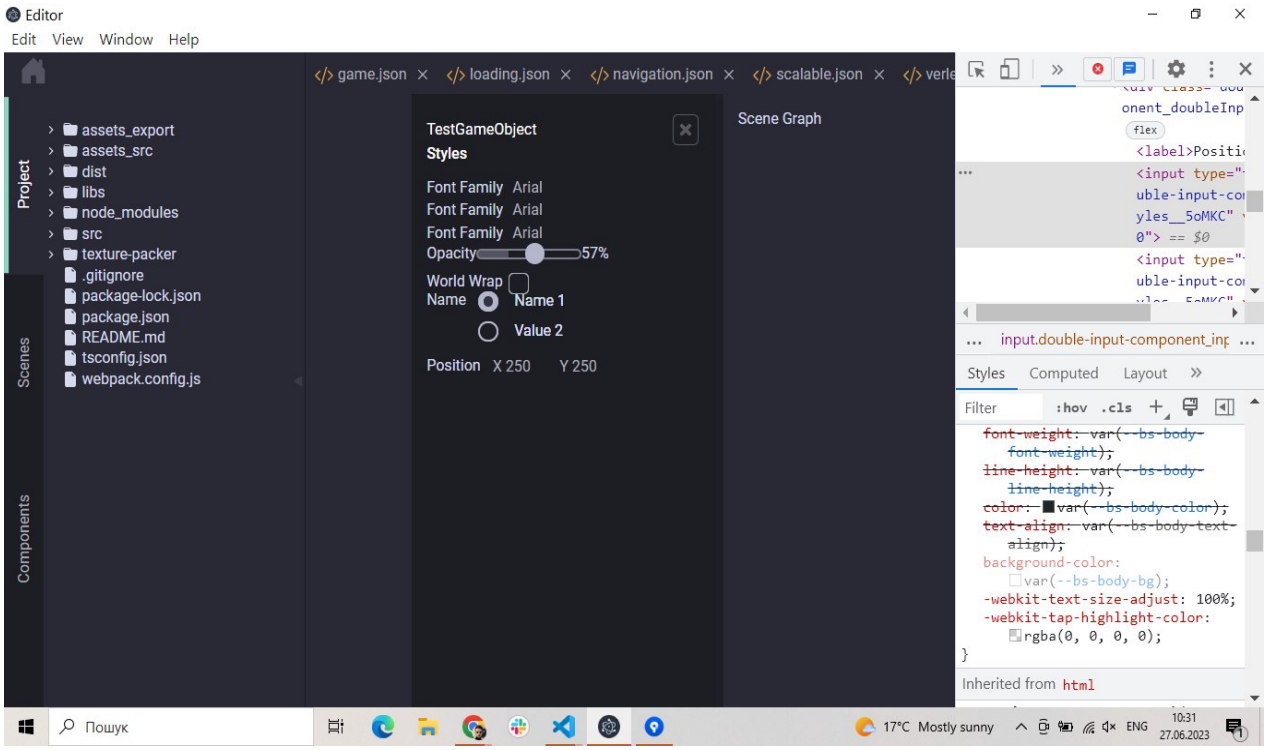

Рис. 4. Прототип інтерфейсу власного редактору

Таблиця 1

Порівняння власного редактору з аналогами

Редактор	Платформа	Мова / Технології	Основні особливості	Недоліки
Tiled	Десктоп (Qt)	C++	Підтримка тайлів, інтеграція з рушіями, JSON/TSX формати.	Важка інтеграція кастомних об'єктів, складна система плагінів
LDtk	Десктоп (MonoGame)	C#	Тайли, автошари, вбудовані правила генерації рівнів.	Вузька орієнтація на платформери, менша гнучкість щодо об'єктів
Ogmo Editor	Десктоп (Electron)	TypeScript	Проста структура, легке збереження в JSON, орієнтований на інді.	Відсутність візуальної ієрархії, обмежена кількість налаштувань
Власна розробка	Десктоп (Electron)	React.js + Pixi.js	Система шарів, drag-and-drop, JSON, undo/redo, відкритість до розширення.	Потребує подальшого розвитку (анімації, інтеграція з рушіями, GUI для подій)

Висновки. У даному дослідженні розглянуто алгоритми, які дозволяють покращити редагування відеоігор, а отже, й покращувати конверсію користувачів та їхню взаємодію з редактором для редагування десктопних відеоігор. Проаналізовано архітектуру та функціональні можливості графічного редактора для редагування сцен у 2D-десктопних відеоіграх. Виокремлено ключові компоненти редактора, що забезпечують зручність, масштабованість та інтеграційність інструменту. Виділено напрями подальших досліджень, такі як: підвищення продуктивності при роботі з великою кількістю об'єктів, додавання складних типів об'єктів (анімації, тригери, логіка), створення системи плагінів для розширення функціональності. Представлені архітектурні підходи можуть бути використані для створення кастомних редакторів, зокрема у сфері інді-розробки ігор. Розглянуто аналоги редакторів для редагування десктопних відеоігор, наведено стислий огляд кожного з них. Крім того, виконано аналіз ігрових рушіїв, які використовуються під час розробки редакторів. На основі отриманих результатів створено прототип власного редактора, основними перевагами якого, зокрема, є кросплатформність завдяки використанню вебстеку, простота масштабування та гнучка архітектура.

Список літератури:

1. Share of game developers working on 2D games worldwide. *Statista*. URL: https://www.statista.com/statistics/1250457/2d-vs-3d-games-indie-developers (дата звернення 25.05.2025)

2. Examples. *Pixi.js*. URL: https://pixijs.io/examples/ (дата звернення 25.05.2025)

3. HTML Drag and Drop API. *Mozilla Developer Network* URL:https://developer.mozilla.org/enUS/docs/Web/API/HTML_Drag_and_Drop_API. (дата звернення 25.05.2025)

4. Unity. URL: https://surl.li/mdhswu (дата звернення 25.05.2025)

5. Unreal Engine. URL: https://skvot.io/uk/blog/ne-soromno-zapitati-yak-pracyuye-unreal-engine (дата звернення 25.05.2025)

6. Lindeijer T. Tiled: A flexible level editor URL: https://www.mapeditor.org (дата звернення 25.05.2025)

7. Bénard S. LDtk – Level Designer Toolkit. URL https://ldtk.io (дата звернення 25.05.2025)

8. Ogmo Editor. Open-source level editor for 2D games URL: https://ogmo-editor-3.github.io (дата звернення 25.05.2025)

Maidaniuk V.P., Skladaniuk O.O. GRAPHICAL EDITOR FOR DESKTOP VIDEO GAME SCENE EDITING

The article presents an analytical review of methods and tools for editing scenes in 2D video games. The relevance of developing such a tool is substantiated, considering the popularity of indie game development and the need for intuitive editors. The technical aspects of implementing functionality such as interactive object movement (drag-and-drop), real-time editing of scene object properties, and layer and scene component management are analyzed. Approaches to storing the scene structure in JSON format are considered, allowing for easy export and import of game level data. Special attention is given to the use of the 2D graphics library Pixi.js as the rendering engine, as well as to architecture design principles based on an object-oriented approach. An analysis is carried out on the use of an event-driven model to ensure interaction between the editor's components. The article also explores the potential for expanding the editor's functionality, including implementing a grid for object alignment, an undo/redo system, support for multiple scenes, and tools for creating more complex game mechanics. Directions for further research are outlined, particularly the development of an adaptive plugin system to extend the editor's features. The main scientific and practical result of the research is the identification of key components necessary for building an effective 2D scene editor, as well as the implementation of a basic version of the editor that can serve as a foundation for future development. The proposed editor's key advantages include a modular interface, ease of project scalability, simplicity of development and maintenance, JSON-based scene saving, a layer system, drag-and-drop editing, basic undo/redo support, integration with Pixi.js as a rendering engine, and cross-platform compatibility. The practical value of the work lies in the possibility of using the developed tool for building prototypes and game levels without the need to write code, thereby accelerating the 2D game development process and allowing developers to focus on game logic and design rather than the technical implementation of scenes. The proposed editor can be integrated into educational programs to familiarize students with the principles of game environment construction and graphics programming.

Key words: *graphic editor, video games, scenes, 2D, drag-and-drop, Pixi.js, JSON, undo/redo, layers, object-oriented programming.*