Біла Л. А., студентка
Житомирський державний університет імені Івана Франка
Науковий керівник – к. пед. н., доцент Мосіюк О. О.

ВАЖЛИВІ АСПЕКТИ СТВОРЕННЯ 2D ПЛАТФОРМЕРА

Індустрія інтерактивних розваг і комп'ютерних ігор розвивається із шаленою швидкістю. Їх розробкою займаються як великі корпорації, так і окремі розробники, які бажають втілити у проекті відеогри свої ідеї та бачення світу. Останнім часом найбільш динамічно розвиваються такі ігрові жанри як: battle royale, moba, rpg, «стратегії», «шутери» та «платформери» [1]. Більшість індивідуальних розробників надають перевагу саме останньому з перерахованих жанрів, адже він є більш простішим у реалізації в порівнянні із іншими.

В класичному 2D платформері умовний персонаж, уникаючи перешкод, рухається зліва направо по певним «платформам», здобуває бали або інші предмети для того, щоб завершити рівень. Таким іграм притаманна мультиплікаційна графіка, а герої зазвичай мають дивні, іноді навіть нереалістичні форми.

Слід зазначити, що серед платформерів набули популярності ще й тривимірні проекти та нескінченні бігові ігри, основна ідея останніх в тому, щоб «пробігти» якомога більшу дистанцію, поки персонаж не втратить всі свої умовні «життєві сили». В основному, вони призначаються для мобільних телефонів та планшетів.

При створенні платформерів досить часто використовують спеціалізовані програми, які мають уже інтеграцію програмної та графічної частини – ігрові рушії. До найпопулярніших варто віднести: Unity, Unreal Engine, Godot, Phaser тощо [6]. Коротко охарактеризуємо їх.

Unity – одне з найпопулярніших середовищ розробки відеоігор. Програми, які дозволяє створювати рушій, працюють на більш ніж 20 платформах (мобільні пристрої, ПК, ігрові консолі та інші) [7]. Інтерфейс середовища є досить привабливим та зрозумілим, саме тому Unity надають перевагу як розробники, які тільки починають розвиток своєї кар'єри, так і досвідчені фахівці. Unity має власну скриптову мову – UnityScript, а також підтримує мову програмування C#.

Unreal Engine є основою багатьох популярних сьогодні ігор рівня AAA [8]. Таке широке використання в цьому секторі пояснюється тим, що рушій був розроблений спеціально для вирішення багатьох складних завдань опрацювання тривимірної графіки. Програмний комплекс є відкритим, а це означає, що його можуть використовувати у різних задачах, зокрема і в кінематографі та архітектурній візуалізації.

Unreal Engine надзвичайно вимогливий до графічних ресурсів, а отже не завжди студенти і ті, хто починає вивчати підходи до створення ігор, можуть дозволити потужний комп'ютер для коректної роботи із ним. Крім того, варто зазначити, що за допомогою Unreal Engine можна створювати 2D платформери, але більшою мірою він розрахований на розробку 3D ігор та просторових динамічних візуалізацій.

Ігрові рушії Godot [2] та Phaser [3] є також вільнопоширюваними програмними комплексами, але вони поступаються можливостям, які надають вище зазначені програми. У той же час вони вимагають менше апаратних ресурсів персональних комп'ютерів у порівнянні із Unity та Unreal Engine.

Важливим елементом розробки платформера є створення певної локації – рівня, по якому буде переміщуватися персонаж. Існує два основних підходи до їх створення. Перший передбачає розробку певної локації спеціальним фахівцем, ще на етапі конструювання, другий – випадкова генерація рівнів за допомогою спеціального алгоритму. У ігровій індустрії кожен із них має своє місце та значення і використовується достатньо часто. Зосередимо увагу на останньому.

Одним із класичних підходів є використання функцій для генерації випадкових чисел, але такий підхід може привести до отримання згенерованих рівнів, які не можливо «пройти» або виконати на них певні задачі. То ж така форма синтезу певного рівня не завжди є

доречною. Для вирішення цієї задачі використовують ряд алгоритмів: конструктивні генератори, алгоритми засновані на пошуку та комбіновані [4].

Перший тип – алгоритми конструктивних генераторів – є надзвичайно простими. Такий підхід передбачає створення певного умовного – агента, який буде імітувати рух ігрового персонажа (біг і стрибки). Агенту дозволяється випадково рухатися і стрибати на порожньому рівні, у той час як інша функція фіксуватиме всі його рухи та зупинки. На основі записаного шляху програма додаватиме необхідні платформи та інші об'єкти у порожній рівень. Такий підхід легко відповідає умові відтворюваності, однак є певні сумніви, наскільки утворені рівні будуть різноманітні. Прикладом такого генератора є Rhythm-Based Level Generator, який описаний у статті за авторства Сміта Г. [5].

Алгоритми на основі пошуку базуються на зовсім іншій ідеї. Основним принципом таких алгоритмів є пошук у просторі всіх існуючих рівнів і вибір найкращого за завчасно вказаними критеріями. Досить часто пошук виконується за допомогою техніки машинного навчання (наприклад еволюційного алгоритму), у якому визначаються критерії, що виявляють, наскільки якісно створений певний рівень.

Ну і останній варіант – комбінований, який поєднує переваги першого та другого підходів, але вимагає більше часу для реалізації.

Висновки. Для того, щоб створити власну гру, потрібно визначитись з жанром та з програмною мовою, яка вам найбільше імпонує (важливо, щоб мова програмування підтримувалася ігровим рушієм також). При виборі середовища бажано звертати увагу на його вартість, доступність, легкість у використанні, набір інструментів та можливості, які він надає.

Вибір алгоритму для створення випадкового рівня платформера має відбуватися із урахуванням особливостей вимоги до створюваної гри, складності реалізації та масштабності проекту.

Список використаних джерел

1. Ігрові жанри. URL: https://training.qatestlab.com/blog/technical-articles/games-genres/ (дата звернення 28.04.2022).
2. Godot. URL: https://godotengine.org/ (дата звернення 28.04.2022).
3. Phaser. URL: http://phaser.io/ (дата звернення 28.04.2022).
4. Potoček T. Level generation techniques for platformer games. URL: https://www.tobice.cz/publications/level-generation-techniques-for-platformer-games.pdf (дата звернення 28.04.2022)..
5. Smith G., Treanor M., Whitehead J., Mateas M. Rhythm-based level generation for 2d platformers. In Proceedings of the 4th International Conference on Foundations of Digital Games, pages 175–182. ACM, 2009.
6. Top 10 Game Development Engines in 2022. URL: https://www.youngwonks.com/blog/Top-10-Game-Development-Engines-Today (дата звернення 28.04.2022).
7. Unity. URL: https://unity.com/ru/products/unity-platform (дата звернення 28.04.2022).
8. Unreal Engine. URL: https://www.unrealengine.com/en-US/ (дата звернення 28.04.2022).