УДК 004.514 DOI: https://doi.org/10.30837/IYF.IIS.2024.375

ВИКОРИСТАННЯ СТРУКТУР ДАНИХ ДЛЯ РЕАЛІЗАЦІЇ ЧЕРГИ ПОДІЙ В ІГРАХ З МЕХАНІКОЮ ОБМЕЖЕННЯ ЧАСУ

Бухало В. О.
Науковий керівник – ст. викл. Новіков Ю. С.
Харківський національний університет радіоелектроніки, каф. ПІ
м. Харків, Україна
e-mail: volodymyr.bukhalo@nure.ua

The current work investigates the use of data structures in developing program code for managing time and events in a gaming environment, where the execution of a particular action is time-constrained. Creating such a tool involves implementing a basic data structure of a queue and modifying it to track and control each player's time, manage the sequence of turns, and respond to events and actions of players within a specified time period. Using the data structure of a queue, the developed software module ensures efficient execution of events in the game process. The work utilized the Node.js runtime environment and the TypeScript programming language.

Ігрова механіка – система або моделювання, заснована на правилах, які полегшують і спонукають користувача досліджувати та вивчати властивості свого простору можливостей за допомогою механізмів зворотного зв'язку [1].

Обмеження часу – це ігрова механіка, яка вимагає від гравця досягнення мети протягом певного періоду часу [2]. Наприклад, ця механіка часто використовується в багатокористувацьких командних іграх в межах однієї кімнати (same-room multiplayer party games), де кожна міні-гра чи подія в ній йде одна за одною і триває протягом певного часу, який регулюється або таймером, або тригером завершення від гравця. Для реалізації таких ігор необхідно мати ефективний механізм для управління чергою подій та часом кожної події окремо.

Дана робота спрямована на реалізацію програмного коду, призначеного для використання в межах ігрового додатку, що забезпечує ефективне керування порядком множини ігрових подій з обмеженим часом їх існування. Для реалізації програмного модулю використовується черга як структура даних для управління ходом гри та змінами стану.

Програмний модуль забезпечує наступні можливості:

– налаштування тривалості виконуваної дії;

– налаштування функцій оберненого виклику;

– створення черги подій;

– зміну виконуваної події у реальному часі при тригерах, ініційованих гравцями;

– додавання як окремих, так і групових подій до черги (у разі, якщо елемент черги, який обслуговується, містить декілька подій, то всі вони виконуються асинхронно);

– можливість контролю стану виконуваних подій.

Створення подібного програмного коду досягається завдяки використанню вбудованих інструментів програмної платформи Node.js та мови програмування TypeScript, без використання готових бібліотечних рішень. Налаштування тривалості виконання функцій реалізується за допомогою вбудованої функції «setTimeout», яка приймає у якості аргументу функцію, що має бути виконана, та часовий інтервал, протягом якого вона буде викликана. Ввесь період, що пройшов до фактичного виконання функції, є періодом, у якому вона вже виконується. Для ефективної реалізації черги подій використовується структура даних «черга», яка базується на імплементації масиву.

Реалізовано об’єктно-орієнтований клас, який включає поля для відображення подій, їх черги та поточних виконуваних дій. Після завершення виконання поточних подій, наступний елемент із черги переміщується у поточні події, і їх виконання розпочинається знову. Цей процес триває доти, поки черга не стане порожньою або не буде перервана через дії гравця. Події налаштовані таким чином, що можуть включати в себе різні дії, кожна з яких має свій час виконання та власні функціональні можливості. Використання саме об’єктно-орієнтованої парадигми пояснюється тим, що в розробці ігор широко використовується саме вона, оскільки дозволяє легко моделювати об’єкти гри, їх взаємодію та стан, що сприяє структурованості та розширюваності коду.

Принцип роботи програмного коду графічно наведено на рисунку 1.

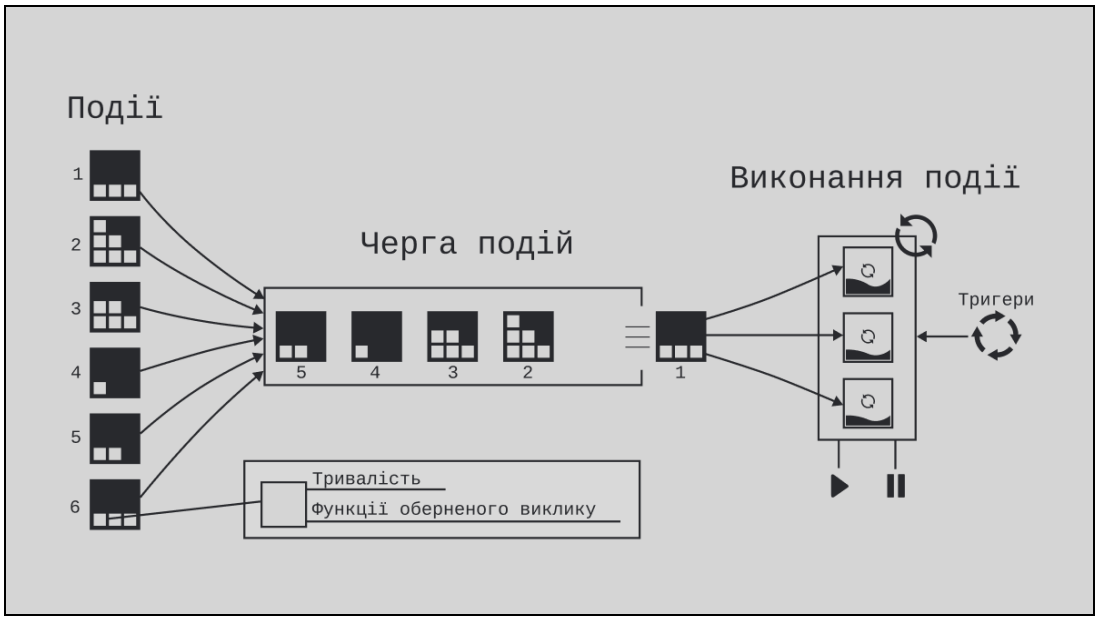

Рисунок 1 – Графічне представлення програмного коду

Черга використовується для забезпечення послідовного виконання подій відповідно до заданих часових затримок. Черга діє як структура даних FIFO (перший прийшов, першим пішов) [3], забезпечуючи керування порядком виконання подій. Коли події додаються до черги, вони виходять з неї та обробляються синхронно, даючи гарантію, що дії обробляються в жорстко визначеному порядку, запобігаючи можливі конфлікти або розходження в логіці гри. Однак, логіка подій, які виходять з черги виконується асинхронно, що дозволяє виконувати дії кожної події незалежно від інших, що особливо важливо багатокористувацьких командних іграх в межах однієї кімнати, де гравці можуть взаємодіяти з різними елементами гри одночасно.

Таким чином розроблений програмний код забезпечує ефективне керування часом та подіями у ігровому середовищі з обмеженим часом їх існування. Використання черги дозволяє забезпечити послідовність виконання подій. При цьому розроблений код здатний відстежувати стан подій та реагувати на тригери від гравців, що є важливим аспектом ігрового досвіду.

Список використаних джерел:

1. Часовська А. О. Застосування ігрових механік в комп’ютерних відеоіграх / А. О. Часовська // Радіоелектроніка та молодь у XXI столітті: тези доповідей 27-го Міжнародного молодіжного форуму, 10–12 травня 2023 р. Харків : ХНУРЕ, 2023. Т. 3. С. 155–156.

2. Time limit – TheAlmightyGuru. TheAlmightyGuru.com. URL: http://www.thealmightyguru.com/Wiki/index.php?title=Time_limit#:~:text=A%20time%20limit%20is%20a,a%20way%20to%20add%20tension. (дата звернення: 21.02.2024).

3. What is a FIFO queue and you to implement it efficiently. IME-USP – Instituto de Matemática e Estatística da Universidade de São Paulo. URL: https://www.ime.usp.br/~pf/algorithms/chapters/queues.html (дата звернення: 21.02.2024).