УДК 004.415.53:004.421

ОПТИМІЗАЦІЯ ЛОГІКИ ЧЕРЕЗ РОЗПОДІЛ EVENT TICK, TIMER EVENTS ТА DELEGATE EVENTS В UNREAL ENGINE 5

Ханьжин М.А., Новіков Ю.С.
e-mail: mykyta.khanzhyn@nure.ua, yuriy.novikov@nure.ua
Харківський національний університет радіоелектроніки, каф. ПІ
м. Харків, Україна

The purpose of this work is to create three methods of logic updates in Unreal Engine 5: Event Tick, Timer Events, and Delegate Events. The analysis is based on tests demonstrating the performance of each method under different conditions. In my research, I tested them using the example of rotating a table, which allowed me to observe how the methods behave during continuous object transformations. Additionally, I examined their efficiency when introducing complex computations to increase CPU load. The results highlight the advantages and limitations of each approach, providing insights into optimizing logic updates for better performance in Unreal Engine 5 projects.

Швидкий розвиток ігрової індустрії вимагає ефективного використання ресурсів CPU. В Unreal Engine 5 для реалізації логіки найчастіше використовується Event Tick [1], що збільшує навантаження на процесор. Альтернативними методами є Timer Events [2], який виконує обчислення через задані інтервали часу, та Delegate Events [3], що активується лише при зміні введення. Аналіз їхньої продуктивності допоможе обрати оптимальний підхід для стабільної роботи проєктів.

Для дослідження створено сцену з алгоритмом обертання столу, що реагує на введення користувача. Під час перетягування мишею значення осей множаться на швидкість обертання та застосовуються через Add Relative Rotation (див. рис. 1). Якщо введення припиняється, обробка зупиняється. Логіка реалізована трьома методами: Event Tick оновлює її кожен кадр, Timer Events працює за таймером, а Delegate Events активується зміною введення.

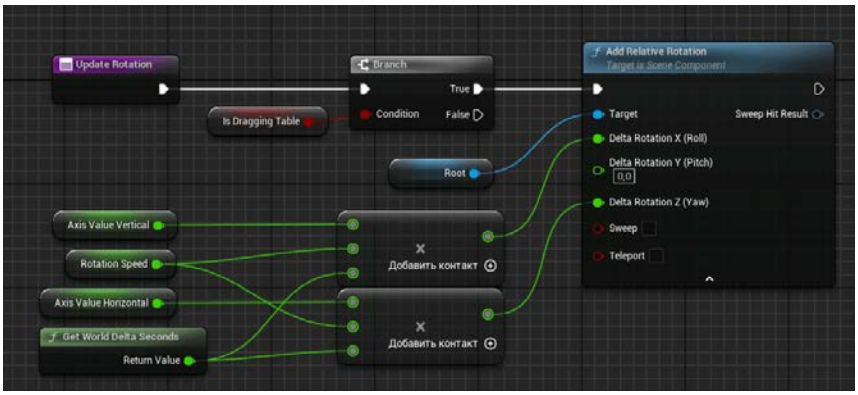

Рисунок 1 – Алгоритм обертання сцени із столом

Тестування без складних обчислень. У цьому сценарії тестувалося лише обертання столу без додаткових обчислювальних операцій. Усі тести мають однакові умови. Приклад тестування (див. рис. 2).

Рисунок 2 – Приклад тестування обертання столу

Виконавши дослідження зафіксуємо значення, додавши їх до таблиці. Результати тестування наведено у таблиці 1.

Таблиця 1 – Продуктивність трьох методів без навантажень

Метод	FPS (макс.)	FPS (мін.)	Frame Time (мс)	Game Thread (мс)	GPU Time (мс)
Event Tick	84	68	12–13	11–13	6–7
Timer	86	68	12–13	11–13	6–7
Delegate	85	69	12–14	11–13	6–7

Без складних обчислень усі методи показали майже ідентичну продуктивність. Поворот столу був плавним у всіх випадках. Це свідчить, що для базових трансформацій, таких як обертання чи переміщення, всі три підходи можуть використовуватися без суттєвих втрат продуктивності.

Тестування зі складними обчисленнями. Оскільки під час базового тестування отримані значення майже не відрізнялися, виникла необхідність у підвищенні обчислювального навантаження. Для цього в кожному з методів було додано виконання математичної формули:

$$\sin 2 + \cos 3 \cdot \tan 4,\ i = \overline{1,10000},$$

де i – номер ітерації циклу, що змінюється в межах від 1 до 10000.

Тестування проводилося за однакових умов для всіх методів. У процесі експерименту було отримано результати для Event Tick та Delegate Events, проте при використанні Timer Events виникла помилка, пов’язана з недостатньою частотою оновлення. Для коректної роботи інтервал таймера було збільшено до 0,05 с, однак це призвело до зниження плавності обертання об’єкта.

Виконавши дослідження зафіксуємо значення, додавши їх до таблиці. Результати тестування наведено у таблиці 2.

Таблиця 2 – Продуктивність трьох методів із навантаженнями

Метод	FPS (мін.)	Frame Time (мс)	Game Thread (мс)	GPU Time (мс)	Додаткові спостереження
Event Tick	24	41	41	8	Стабільна робота, хоча FPS значно падає
Timer (0.01 сек)	-	-	-	-	Помилка: «Виявлено нескінченну петлю»
Timer (0.05 сек)	30	30	29	7	Поворот став рваним, але без помилок
Delegate	14	70	70	9	Найнижча продуктивність серед методів

Висновки. Дослідження показало, що продуктивність методів Event Tick, Timer Events та Delegate Events у Unreal Engine 5 залежить від складності завдань. Для простих операцій, як-от обертання столу без складних обчислень, усі методи демонструють подібні результати за FPS, часом кадру та навантаженням на GPU, тож вибір методу у цьому випадку не є критичним.

При складних обчисленнях продуктивність відрізняється. Event Tick забезпечує найкращу стабільність і підходить для завдань із постійним оновленням логіки, наприклад, анімацій чи фізичних симуляцій. Однак при високому навантаженні може знижувати FPS, тому його слід вимикати, якщо він не потрібен. Timer Events демонструє компроміс між продуктивністю та частотою оновлення: інтервал 0,01 сек викликає помилки, 0,05 сек — стабільний, але з втратою плавності. Підходить для логіки, що не потребує оновлення кожного кадру. Delegate Events показує найнижчу продуктивність при частих викликах, але ефективний для рідкісних подій, значно зменшуючи навантаження на систему.

Таким чином, Event Tick доцільно використовувати для плавних анімацій із контролем викликів, Timer Events — для періодичних оновлень, а Delegate Events — для рідкісних подій без шкоди для продуктивності.

Список використаних джерел:

1. Event Tick // Epic Games Documentation. URL: https://dev.epicgames.com/community/learning/tutorials/JPle/unreal-engine-event-tick (дата звернення: 18.02.2025).

2. Using Timers // Epic Games Documentation. URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/using-timers-in-unreal-engine (дата звернення: 18.02.2025).

3. Event Dispatchers and Delegates Quick Start Guide // Epic Games Documentation. URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/event-dispatchers-and-delegates-quick-start-guide-in-unreal-engine (дата звернення: 18.02.2025).