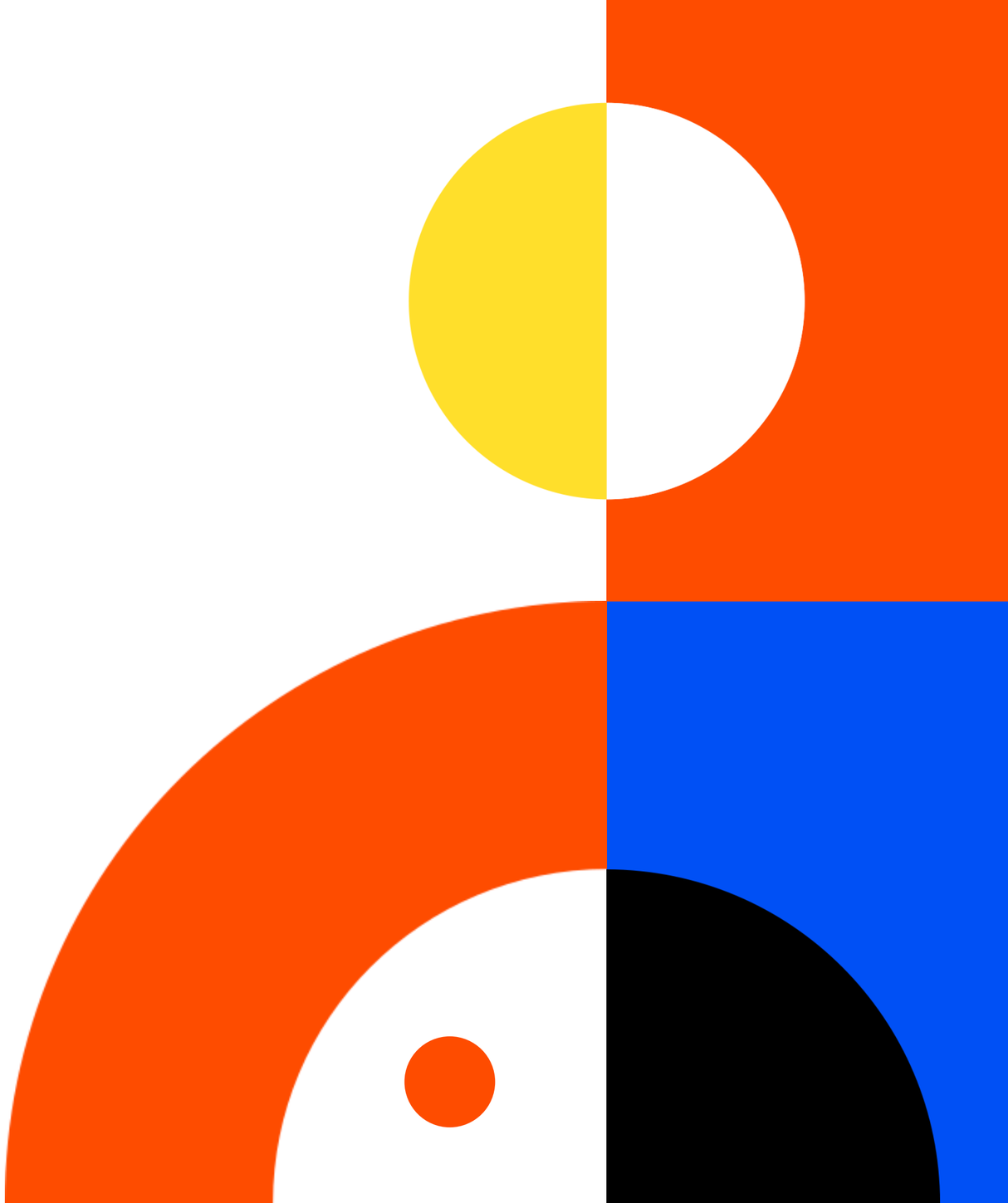
Лекція №7

UI

2025



План

1. Історія розвитку UI в Unity
2. Основні елементи
3. Функціональні елементи

UI

UI (User Interface) — це все, з чим взаємодіє гравець: кнопки, панелі, тексти, індикатори здоров’я, меню тощо.

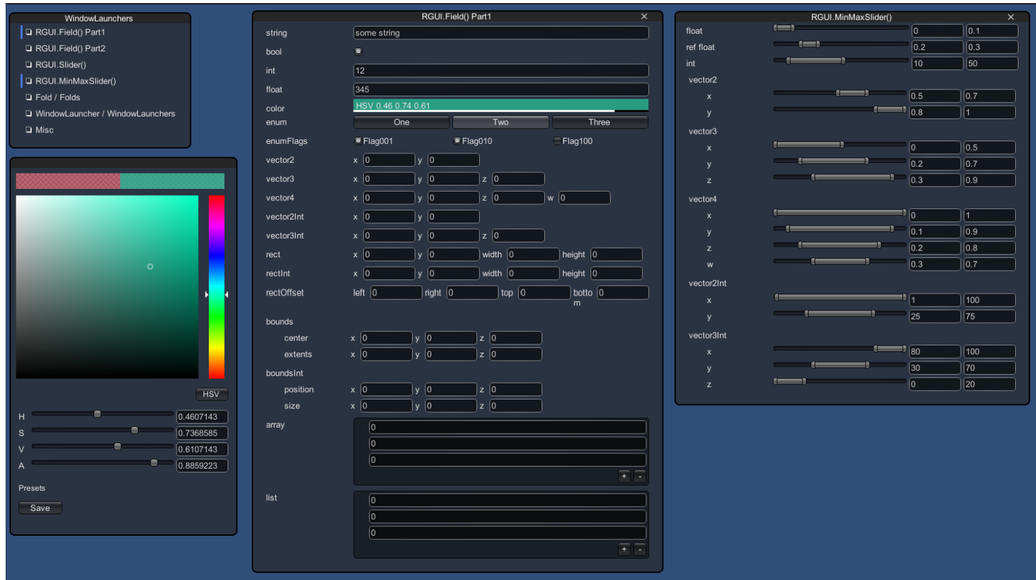
Історія розвитку

Unity пройшла довгий шлях у розвитку своєї системи UI — від примітивних OnGUI() викликів до сучасного UI Toolkit, що нагадує веб-технології (UXML + USS).
Розглянемо етапи еволюції по черзі.

Legacy GUI (IMGUI, або Immediate Mode GUI)

У найперших версіях Unity (до 2014 року) інтерфейс будувався за допомогою **Immediate Mode GUI (IMGUI)**.

```
void OnGUI()
{
    if (GUI.Button(new Rect(10, 10, 100, 30), "Click Me"))
        Debug.Log("Button Pressed!");
}
```



- **Принцип роботи:**
- UI елементи відмальовувались **кожен кадр** (immediate mode rendering).
- Весь UI писався **через код**, без візуального редагування.
- Викликався метод OnGUI(), який формував інтерфейс під час рендеру кадру.
- **Переваги:**
- Простий для дебагу.
- Підходив для інструментів у редакторі (що Unity робить і досі).
- **Недоліки:**
- Високе навантаження на CPU.
- Відсутність підтримки адаптивного дизайну.
- Неможливо створювати складні UI (меню, HUD) без великої кількості коду.

Unity UI (uGUI)

У 2014 році Unity представила **uGUI** — нову компонентну систему UI, побудовану на Canvas, RectTransform і подіях.

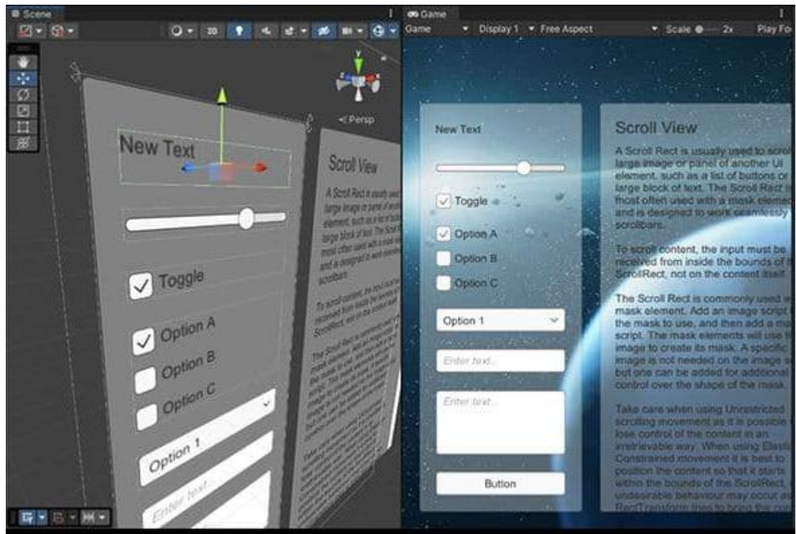
Ключові особливості:

Візуальне редагування прямо у сцені.

Модель основана на подіях (Event System).

Адаптивний дизайн через Anchors.

Підтримка анімацій, Layout Group, ScrollView тощо.



Переваги:

Швидке створення інтерфейсів без коду.

Можна редагувати UI як звичайні GameObject-и.

Підтримує анімації, стилізацію, масштабування.

Недоліки:

Canvas перерендерюється повністю при будь-якій зміні (низька продуктивність у великих UI).

Погано підходить для масивних списків чи “віртуального” UI.

Відсутність повної роздільності між логікою й виглядом.

UI Elements → UI Toolkit

Unity почала розробку **UI Elements**, яка згодом стала **UI Toolkit**.

Основна ідея:

Створити UI подібний до веб-технологій:

- **UXML** — як HTML.
- **USS** — як CSS.

Відокремити представлення (markup) від логіки (C# scripts).

Переваги:

Висока продуктивність, бо не перерендерює весь UI.

Легка стилізація через USS.

Один підхід для **Editor UI** та **Runtime UI**.

Підтримує реактивну архітектуру (Binding).

Недоліки (на ранніх етапах):

Менша кількість компонентів порівняно з uGUI.

Неповна підтримка анімацій і Canvas-подібних можливостей.

Інший підхід до ієрархії та подій, тому не повністю сумісний із старими UI проєктами.

Основні елементи

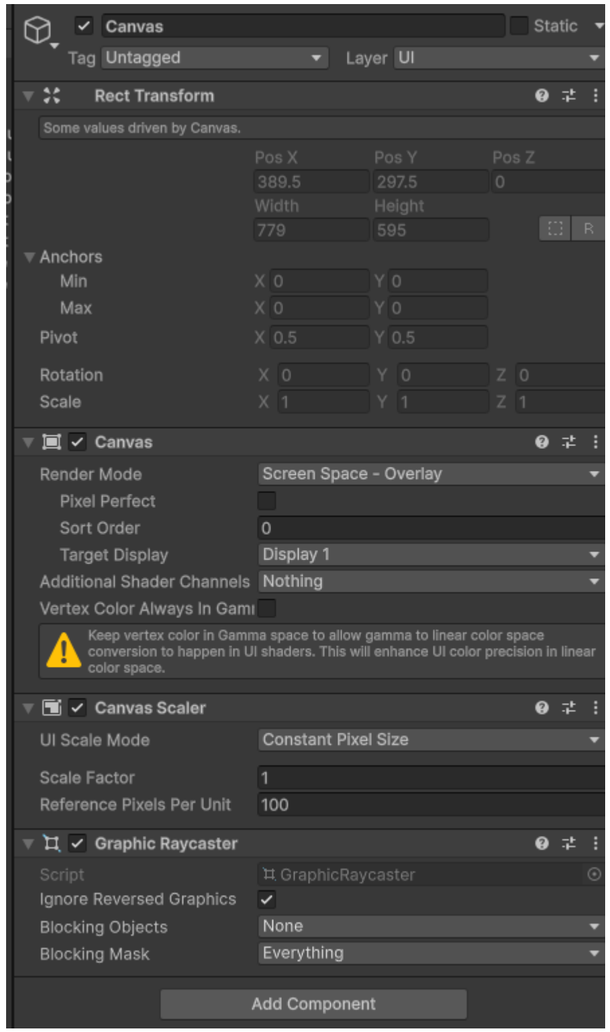
Canvas

Головний контейнер для всіх елементів інтерфейсу.

Визначає, як UI буде відображатись на екрані.

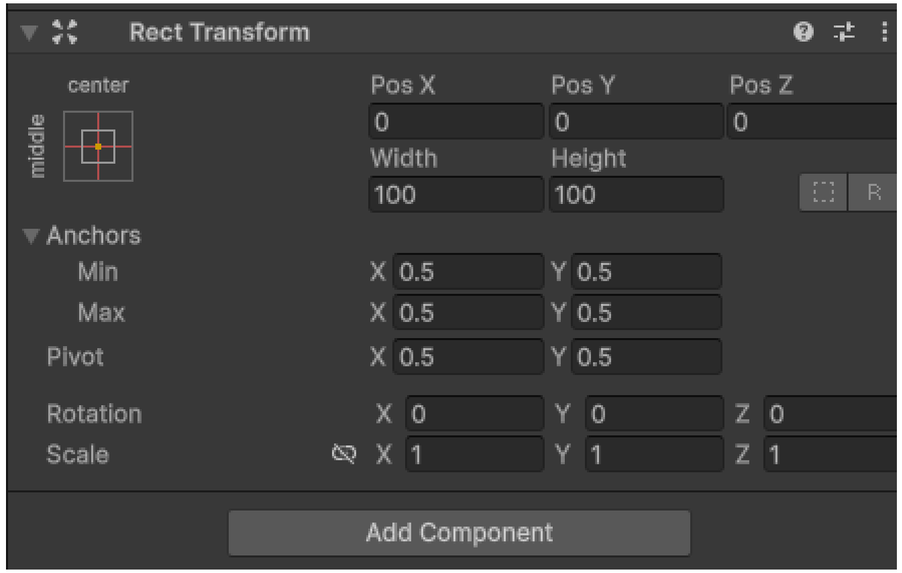
Типи Render Mode:

- **Screen Space – Overlay** — UI поверх усього рендера.
- **Screen Space – Camera** — UI прикріплений до камери.
- **World Space** — UI існує як 3D-об’єкт у сцені.



RectTransform

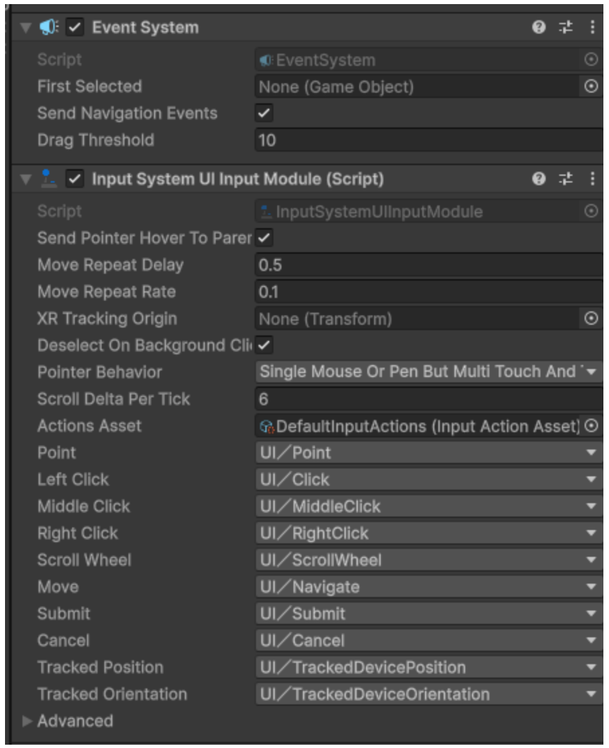
Аналог Transform для UI.
Визначає позицію, розмір і прив’язки.
Має **Anchors** (якорі), які дозволяють адаптувати інтерфейс під різні розміри екранів.
Наприклад, кнопка, прикріплена до правого нижнього кута, залишиться там навіть при зміні роздільної здатності.



Event System

Відповідає за обробку подій (натискання, наведення, drag & drop).
Використовує **Standalone Input Module**, **Touch Input Module**, або **Input System UI Module**.

Без Event System жодна кнопка не буде працювати.



Функціональні елементи

Компонент	Призначення
Text (TMP)	Відображає текст. Використовуйте **TextMeshPro** для кращої якості.
Image	Показує спрайт або текстуру.
Button	Викликає дію при натисканні.
Toggle	Перемикач (on/off).
Slider	Повзунок (напр. гучність).
Scrollbar	Керує прокруткою контенту.
Dropdown	Меню з вибором елементів.
InputField (TMP)	Ввід тексту користувачем.