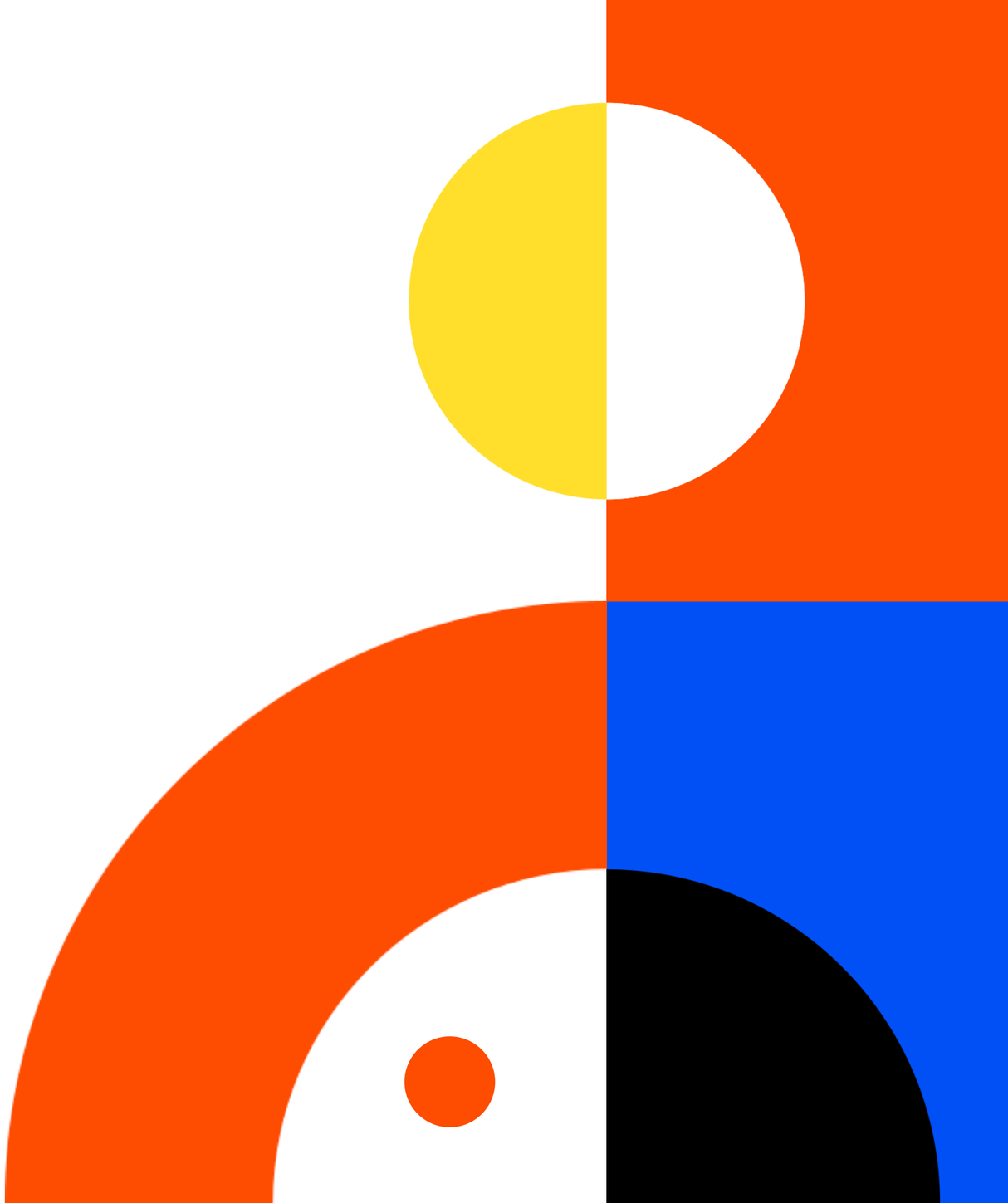
Лекція №5

Налаштування анімацій

2025



План

1. Анімації в Unity. Animator
2. Створення рівня за допомогою TileMap
3. SpriteAtlas

Анімації в Unity. Animator

Система **Animator** є серцем керування анімаціями в Unity. Вона дозволяє створювати складні переходи між різними станами (наприклад, "Спокій", "Ходьба", "Стрибок").

Ключові компоненти

1. **Animator Component.** Прикріплюється до об'єкта, який має анімуватися. Це точка входу для системи.
2. **Animator Controller.** Це основний "графік" або скінченний автомат. Він візуально представляє всі стани анімації та умови переходу між ними.
3. **Animation Clips.** Власне файли, що містять ключові кадри (рух, зміна спрайтів, зміна розміру тощо). Вони вставляються як "стани" у Controller.
4. **Parameters (Параметри).** Змінні (Float, Int, Bool, Trigger), які ви використовуєте у коді, щоб повідомити Controller, що потрібно змінити стан.

Логіка переходу

У скрипті ви змінюєте параметр в Animator, і Controller автоматично переходить до нового стану, якщо умови переходу виконані.

Приклад керування аніматором

```
using UnityEngine;
public class PlayerAnimator : MonoBehaviour{
    private Animator animator;
    private float horizontalInput;
    void Start(){ animator = GetComponent<Animator>(); }
    void Update() {horizontalInput = Input.GetAxis("Horizontal");
        // 1. Встановлюємо булевий параметр 'IsRunning'
        // Якщо гравець рухається (горизонтальний ввід не 0), IsRunning = True
        bool isRunning = Mathf.Abs(horizontalInput) > 0.01f;
        animator.SetBool("IsRunning", isRunning);
        // 2. Встановлюємо Float параметр 'Speed‘.
         // Якщо потрібно мати плавніший перехід або використовувати Blend Tree
        animator.SetFloat("Speed", Mathf.Abs(horizontalInput));
        // 3. Trigger для одноразової події (наприклад, атака)
        if (Input.GetKeyDown(KeyCode.Space)){
            animator.SetTrigger("Attack");
        }   }}
```

Приклад керування аніматором

У прикладі:

У Animator Controller ви створюєте параметр IsRunning (Bool).

Ви створюєте перехід між станами "Idle" (Спокій) та "Run" (Біг).

Умова переходу з "Idle" в "Run" — це коли IsRunning дорівнює True.

Умова переходу назад — коли IsRunning дорівнює False.

Створення рівня за допомогою TileMap

TileMap (Тайлова Карта) — це інструмент Unity, спеціально розроблений для швидкого створення 2D-рівнів, особливо для платформерів та ізометричних ігор, використовуючи маленькі зображення-плитки (тайли).

Основні компоненти

Grid (Сітка) - базовий об'єкт, який створює сітку координат у сцені.

Tilemap - Компонент, який знаходиться на дочірньому об'єкті Grid. Він зберігає інформацію про те, який тайл розташований у якій клітинці сітки.

Tile Palette (Палітра Тайлів) - вікно у редакторі (Window -> 2D -> Tile Palette), де зберігаються всі доступні тайли. З неї ви берете тайли і малюєте ними на Tilemap.

Переваги TileMap

- **Швидкість.** Можна створювати великі рівні, малюючи як пензлем.
- **Оптимізація.** Unity автоматично об'єднує колайдери багатьох тайлів в один великий колайдер (TilemapCollider2D), що значно підвищує продуктивність.
- **Шари.** Можна використовувати кілька Tilemap-шарів (наприклад, "Фон", "Платформи", "Передній план"), щоб створювати глибину та складність сцени.

Приклад застосування TileMap

	Опис
Стиль	Super Mario Bros., Celeste, Terraria.
Шари TileMap	1. Платформи (Foreground/Collision) малюються тайли, по яких рухається гравець (земля, цегла, блоки). На цьому шарі додається TilemapCollider2D. 2. Фон (Background) хмари, задні стіни, декорації, які не мають колайдера. 3. Декорації (Details) дрібні елементи, трава, світильники, які малюються поверх платформ, але також не мають колайдера.
Ключова перевага	Unity автоматично генерує єдиний, оптимізований колайдер для всього шару платформ, що значно підвищує продуктивність порівняно з додаванням BoxCollider2D до кожного окремого блоку.

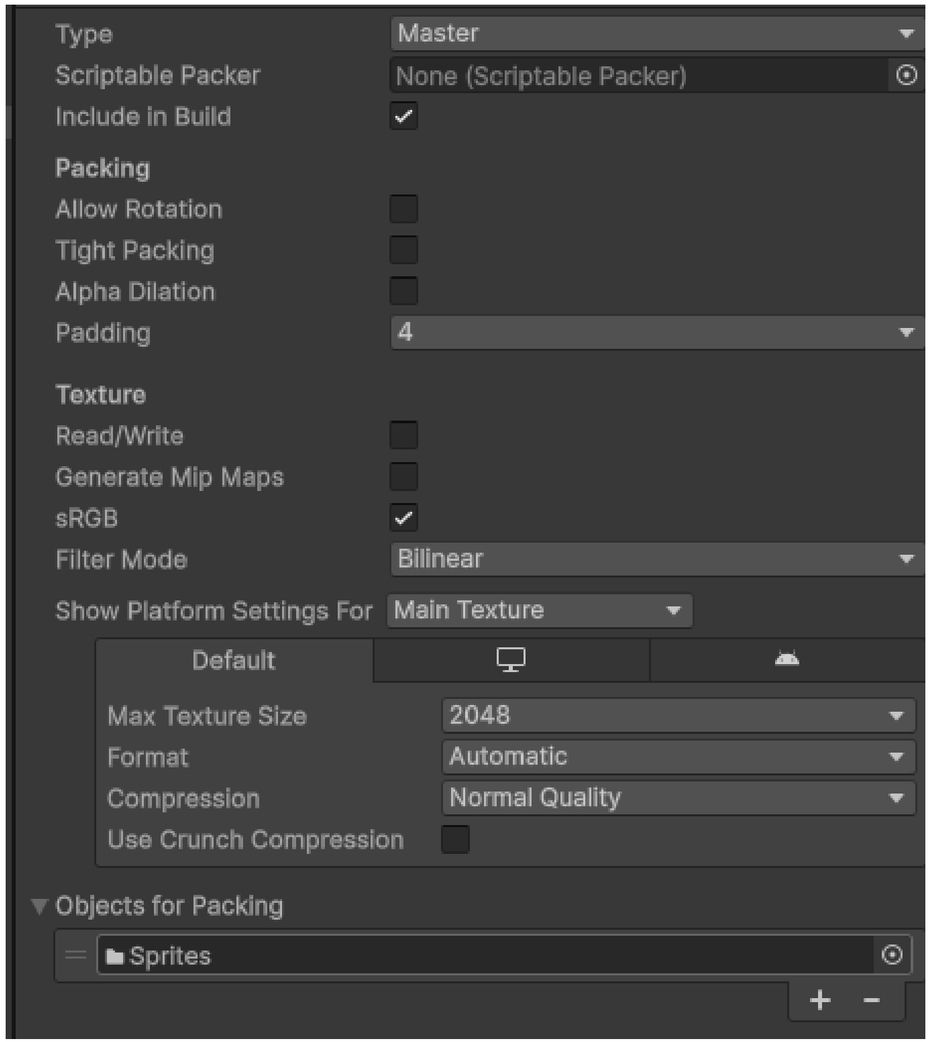
Приклад застосування TileMap

	Опис
Стиль	Ранні RPG (наприклад, Diablo 1), деякі стратегії.
Особливості	Тайли мають ромбоподібну або діагональну форму. Це вимагає спеціального компонента Isometric Tilemap (або Hexagonal Tilemap для гексагональних сіток).
Ключова перевага	Керування порядком (Sorting Order): Завдяки ізометричному TileMap Unity автоматично сортує об'єкти (гравця, ворогів) так, щоб вони правильно перекривали стіни чи меблі, створюючи ілюзію глибини. Об'єкти, що мають нижче положення по осі Y, малюються вище.

Sprite Atlas

- **Sprite Atlas** — це спеціальний ресурс у Unity, який об’єднує багато окремих спрайтів (зображень) в один великий текстурний аркуш.
- Мета — **зменшити кількість переключень текстур (texture switches)** під час рендерингу, що суттєво покращує продуктивність.

Sprite Atlas



- **Створи атлас:**
  Assets → Create → 2D → Sprite Atlas
- **Додай спрайти або цілі папки:**
  У полі “Objects for Packing” перетягни потрібні текстури.
- **Натисни Pack Preview**, щоб переглянути, як Unity упакувала зображення.
- **Compression і Format:**
  - Використовуй стиснення (Compression: Use platform settings)
  - Перевір якість (RGBA32 / ASTC / ETC2 — залежно від платформи)
- **Include in Build:** має бути увімкнено, якщо атлас повинен бути в збірці.