



THIẾT KẾ VÀ PHÁT TRIỂN GAME

Bài 4: GameObject trong Unity



Nội dung

1. GameObject
2. C# Script
3. Làm việc với màn hình console
4. Viết mã tìm hiểu về vòng đời của GameObject



Phần 1

GameObject



GameObject

- GameObject là đối tượng cơ bản trong tất cả các màn hình game của unity
 - Mỗi Game có nhiều Scene (màn hình)
 - Mỗi Scene có nhiều GameObject
- Unity sử dụng cách tiếp cận “phi hướng đối tượng” trong việc xây dựng các đối tượng trong Scene
 - Tất cả các đối tượng con đều là GameObject
 - GameObject là sealed class (không thể được kế thừa)
 - Các kiểu GameObject được làm phong phú và khác nhau bằng cách gắn thêm một hoặc nhiều component với các giá trị thuộc tính khác nhau



GameObject

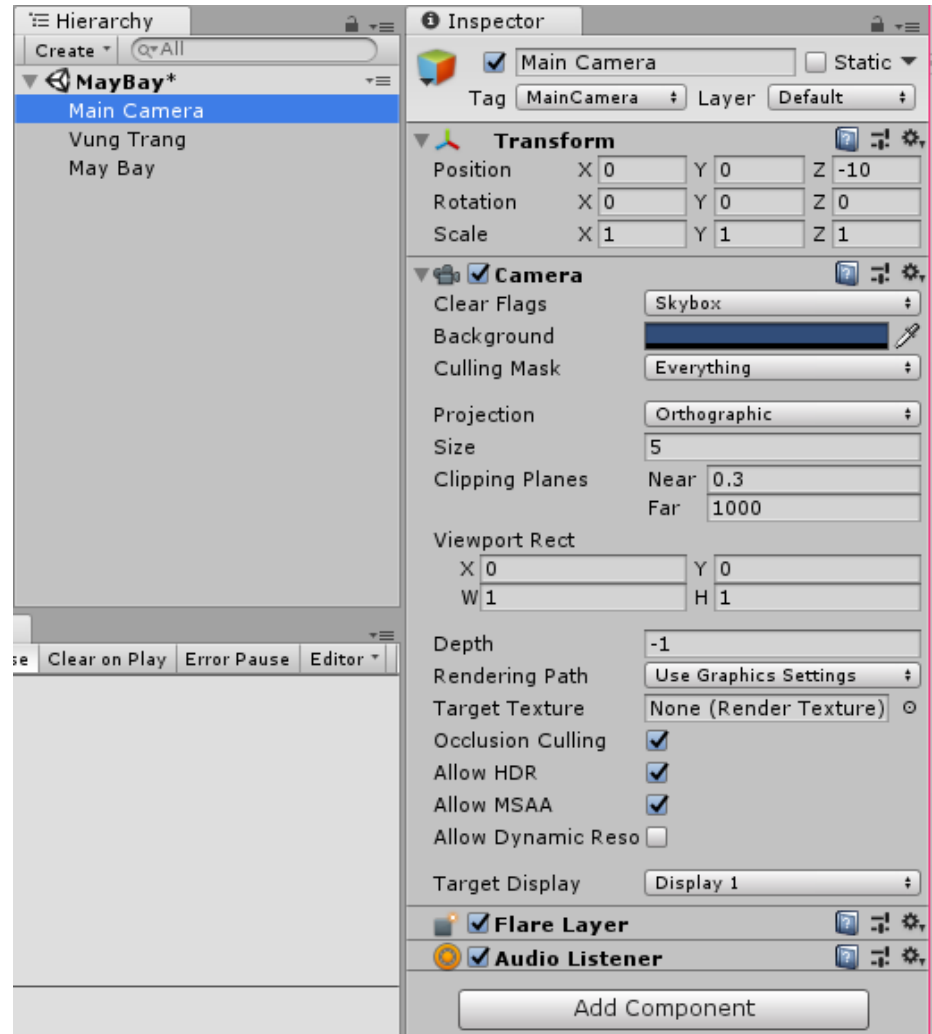
- Một GameObject rỗng, mới được tạo ra
 - Tên là “Vung Trang”
 - Chưa có tag, thuộc layer mặc định
 - Component: Transform



GameObject



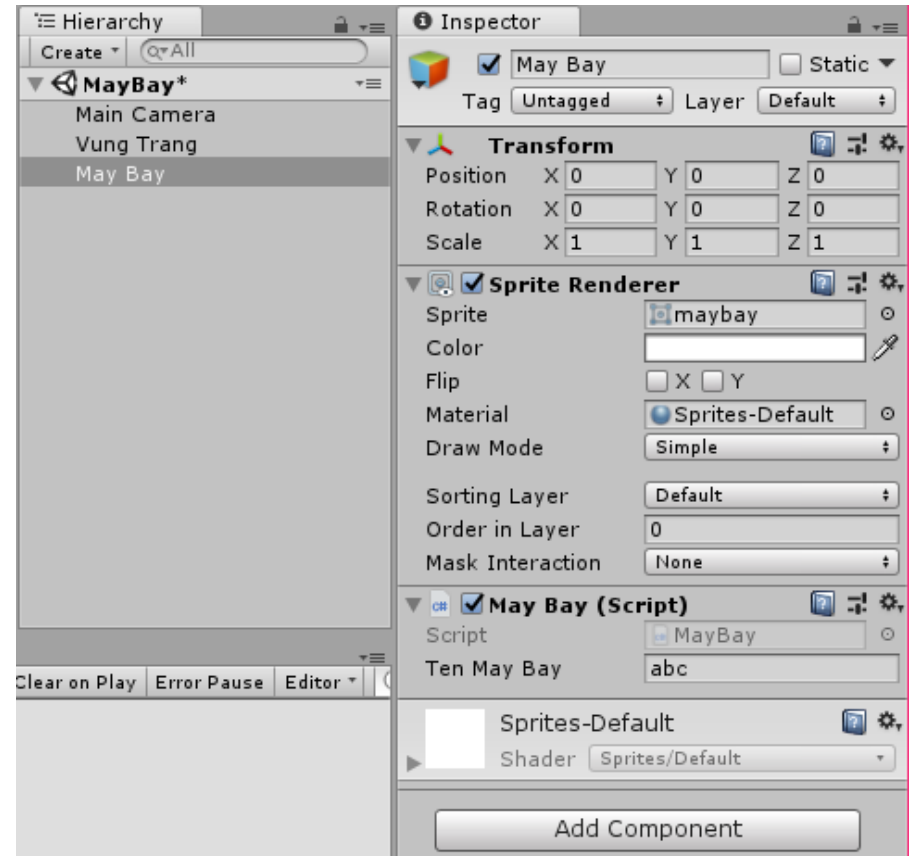
- GameObject “camera”:
 - Tên là “Main Camera”
 - Có tag “MainCamera”
 - Thuộc layer mặc định
 - Component:
 - Transform
 - Camera
 - Flare Layer
 - Audio Listener



GameObject



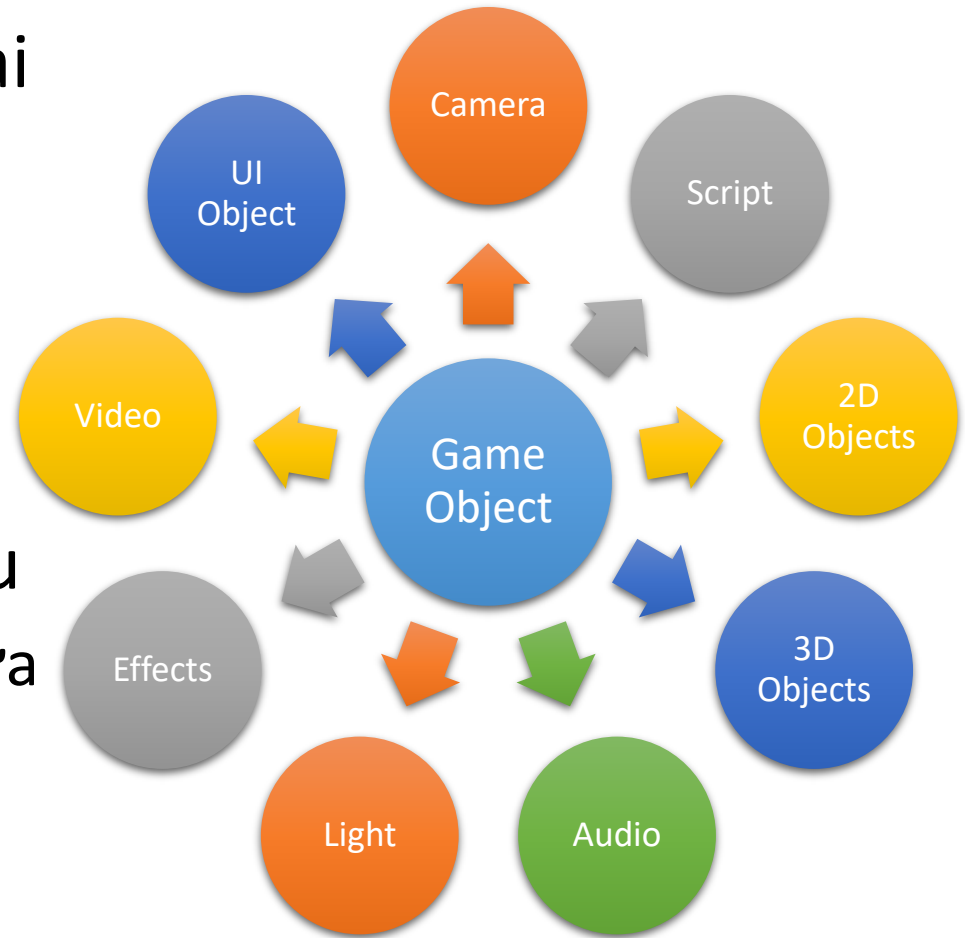
- GameObject “May Bay”:
 - Tên là “May Bay”
 - Chưa có tag
 - Thuộc layer mặc định
 - Component:
 - Transform
 - Sprite Renderer
 - May Bay (Script)
 - Ten May Bay???
 - Sprites-Default???



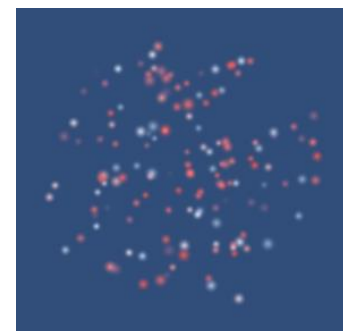
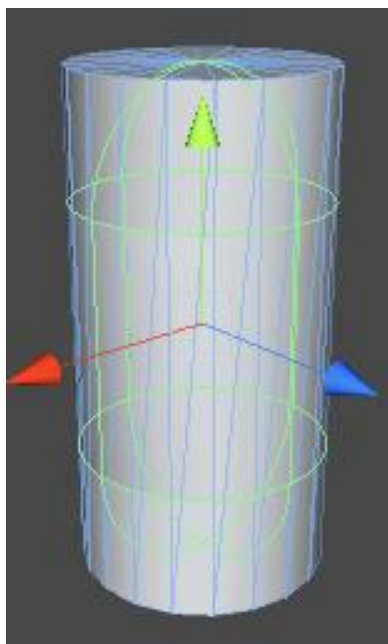
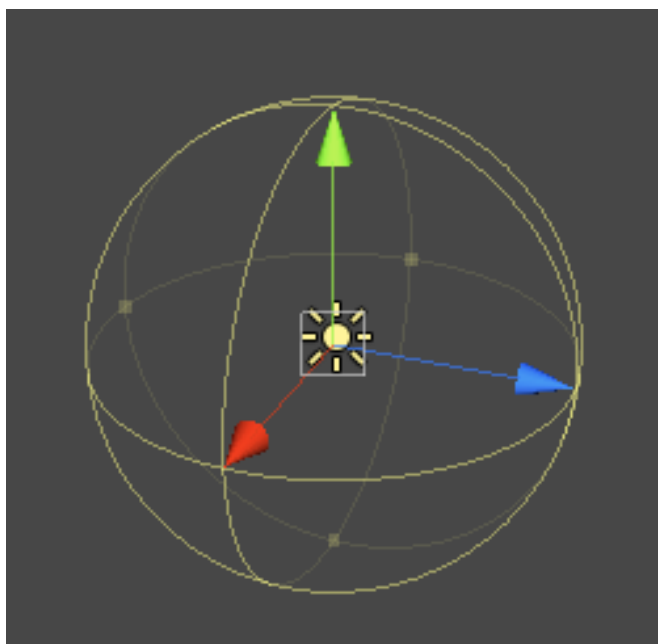
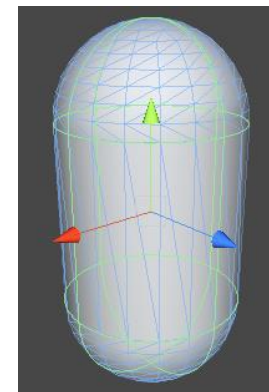
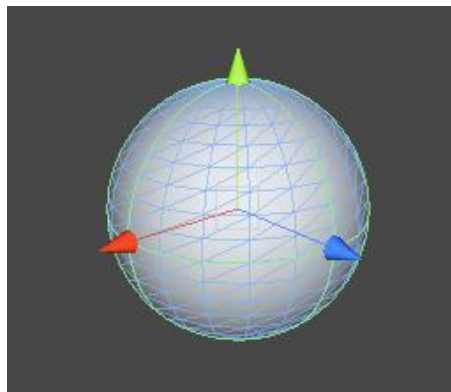
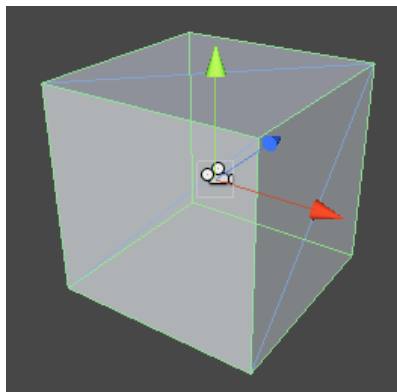
GameObject



- Unity tạo sẵn nhiều loại component
- Bằng việc kết hợp các component, lập trình viên tạo ra các loại GameObject khác nhau
 - Như vậy: không thể đưa một tính năng mới vào game nếu component hỗ trợ nó chưa có



GameObject





GameObject

- GameObject có thể được tạo ra bằng nhiều cách khác nhau:
 1. “By design”: tạo từ đầu trong Scene và thiết lập thuộc tính trực tiếp từ màn hình thiết kế
 2. “By code”: tạo bằng script (tạo một GameObject rỗng rồi thêm các component, tất cả đều bằng code)
 3. “From an instance”: tạo mới bằng cách tạo bản sao của đối tượng đã có và hiệu chỉnh (bằng code)
 4. “From file”: nạp từ Resource (và hiệu chỉnh, tất nhiên)
- Kinh nghiệm: không nên tạo/hủy GameObject quá nhiều, nên tái sử dụng (object pooling)



Phần 2

C# Script



Vai trò của script trong Unity

- Về bản chất Unity coi việc chơi game là quá trình tương tác với các GameObject
- Script là một component đặc biệt
 - Viết bằng C# hoặc Javascript
 - Luôn kế thừa từ MonoBehaviour
- Nhiệm vụ chính của script: mang lại “phần hồn” cho các GameObject
 - Xử lý xem GameObject phản ứng như thế nào đối với các tương tác trong màn chơi
 - Bản thân tên lớp cha đã nói lên điều đó



Vai trò của script trong Unity

- Script C# có thể được viết trong MonoDevelop hoặc Visual Studio hoặc một công cụ tương đương
 - Tất nhiên là nên dùng MonoDevelop
- Kinh nghiệm: nên mở sẵn unity documents vì MonoDevelop hỗ trợ rất kém, đặc biệt với những bạn mới làm quen với unity
- Có thể dùng lẫn script C# và javascript trong cùng một dự án, nhưng cách tham chiếu đối tượng chéo giữa các class khá lộn xộn
 - Không có lý do hợp lý nào cho việc dùng lẫn cả



Phần 3

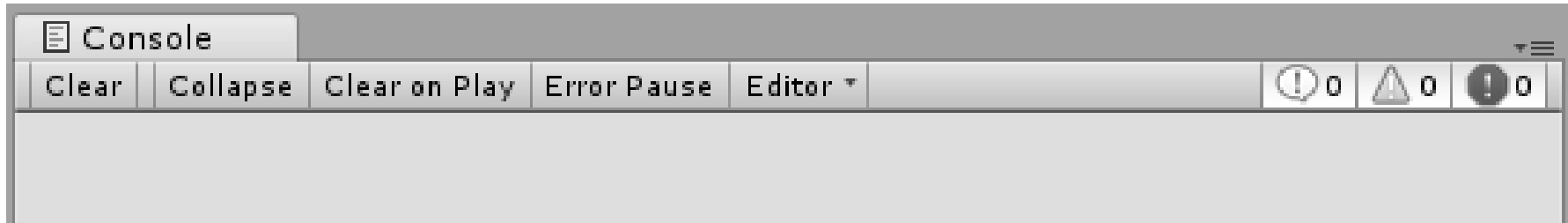
Làm việc với màn hình console

Cửa sổ Console



- Vai trò giống như cửa sổ CMD trong Windows
- Rất quen thuộc với các phần mềm IDE truyền thống
 - Visual Studio, Eclipse, Android Studio,...
- Vai trò chủ yếu để in thông tin gỡ lỗi hoặc cảnh báo
 - Tất nhiên vẫn có những ứng dụng hay sử dụng console
- Console của unity mặc định hỗ trợ unicode
- In thông báo ra màn hình:
 - Dùng phương thức `print` (của `MonoBehaviour`)
 - Dùng `Debug.Log`, `Debug.LogWarning`, `Debug.LogError` tùy vào từng tình huống của game

Cửa sổ Console



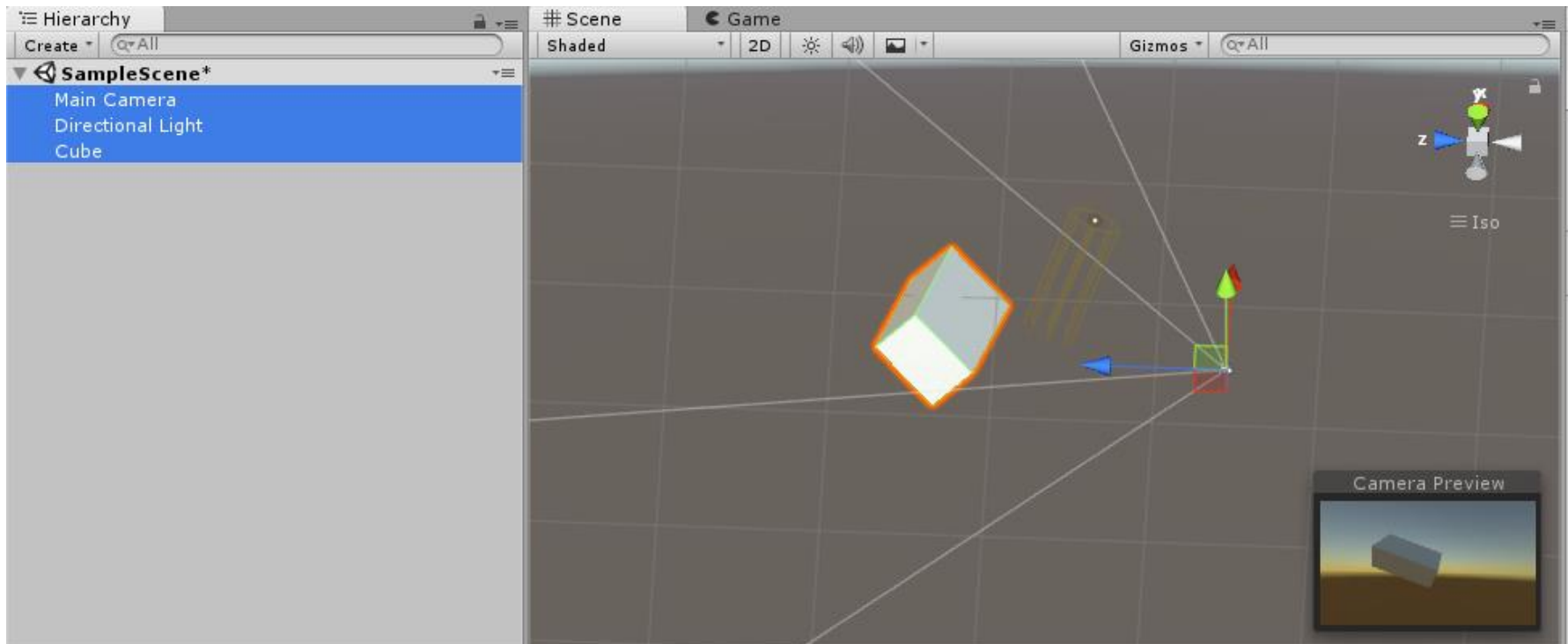
- Một vài chức năng cơ bản:
 - “Clear”: xóa màn hình
 - “Collapse”: thu gọn những dòng giống nhau lại làm một
 - “Clear on Play”: xóa màn hình khi bắt đầu thử game
 - “Error Pause”: dừng khi gặp lỗi
- Có bộ đếm số lần log, warning và error được in ra
- Kinh nghiệm: log ra file khi gặp những tình huống phức tạp



Phần 4

Viết mã tìm hiểu về vòng đời của GameObject

Tạo một scene thử nghiệm





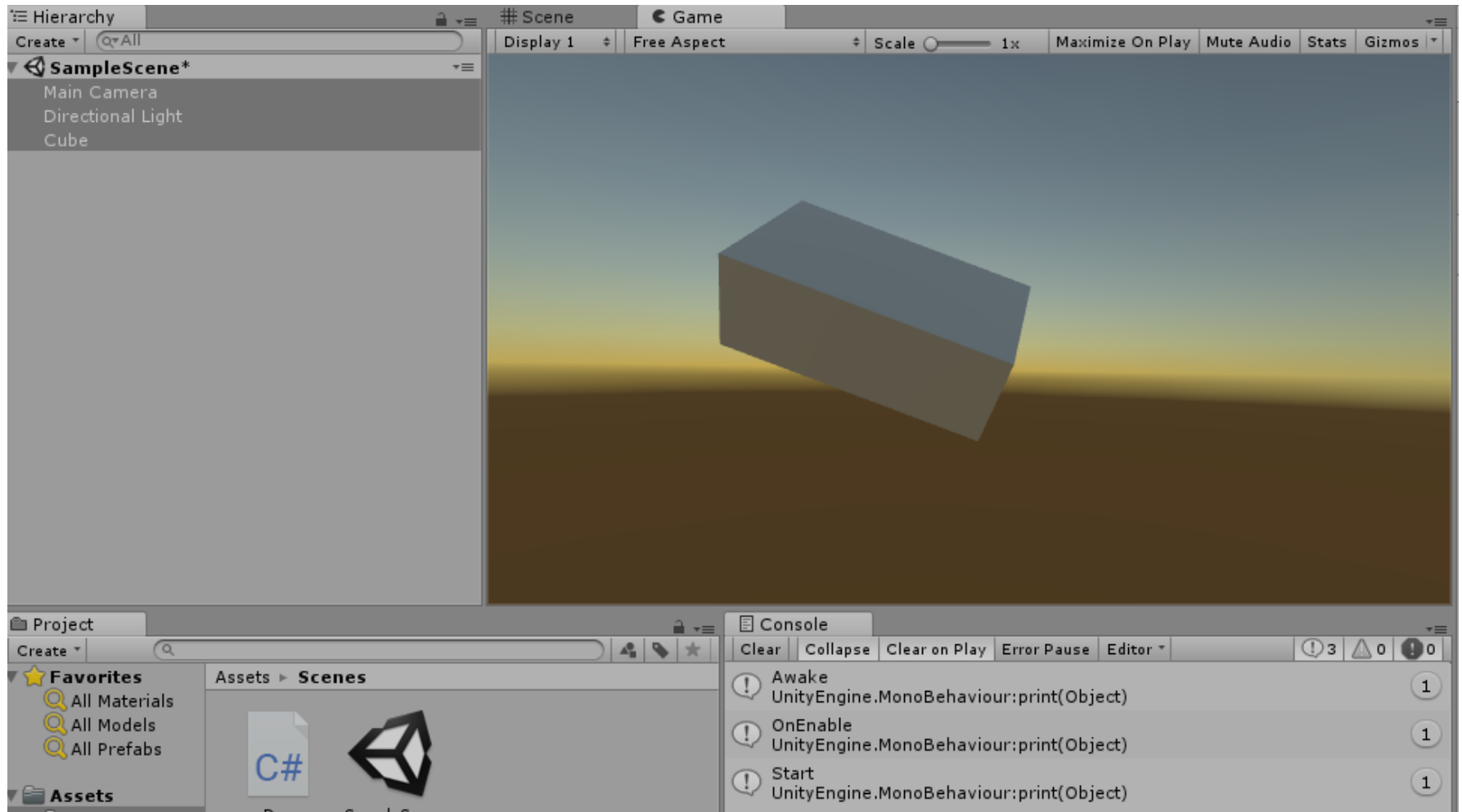
Vòng đời của Game Object

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Den : MonoBehaviour {
    void Awake() { print("Awake"); }
    void Start() { print("Start"); }
    void OnDisable() { print("OnDisable"); }
    void OnEnable() { print("OnEnable"); }
}
```

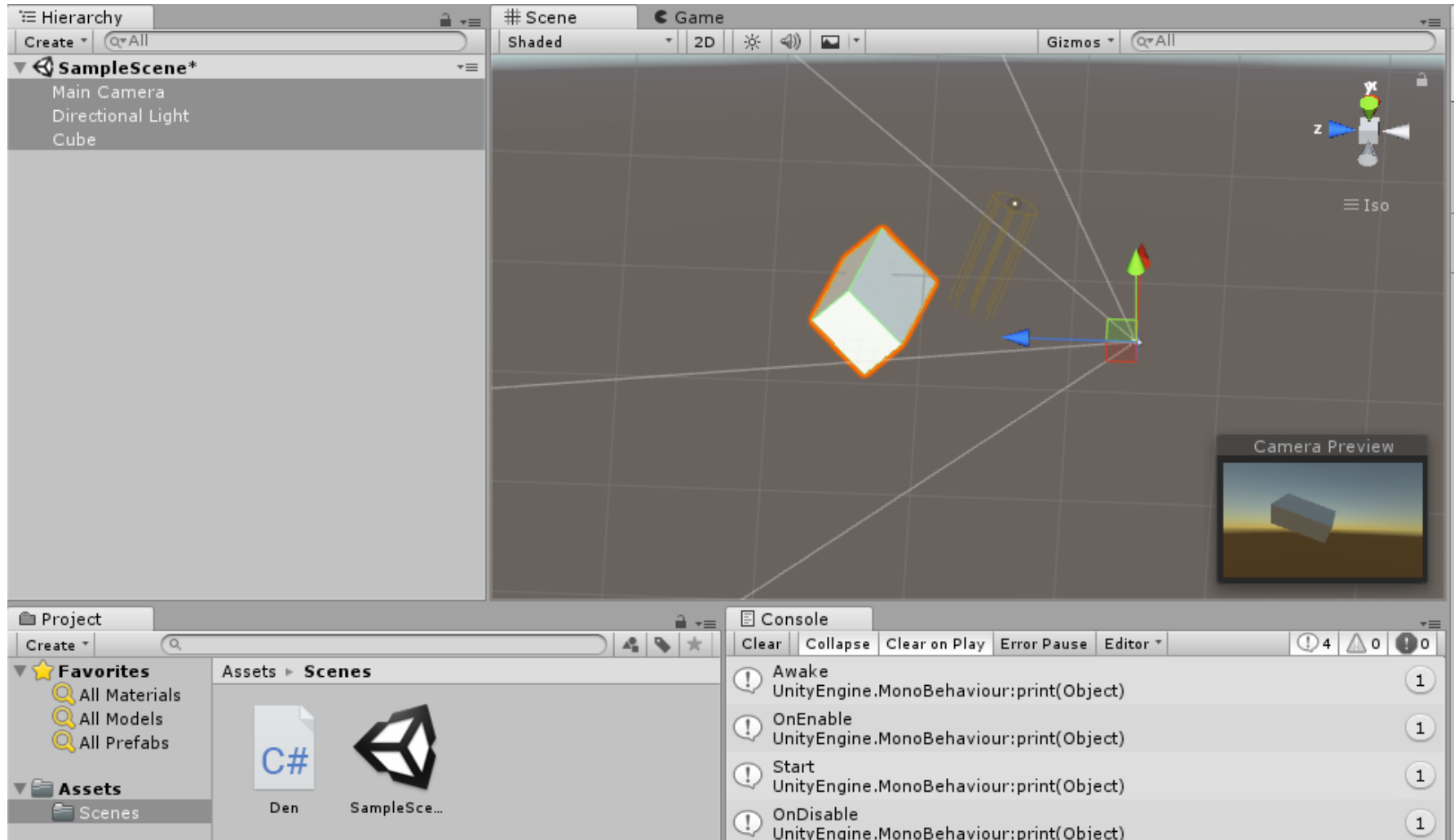


Vòng đời của Game Object





Vòng đời của Game Object





Update vs FixedUpdate

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

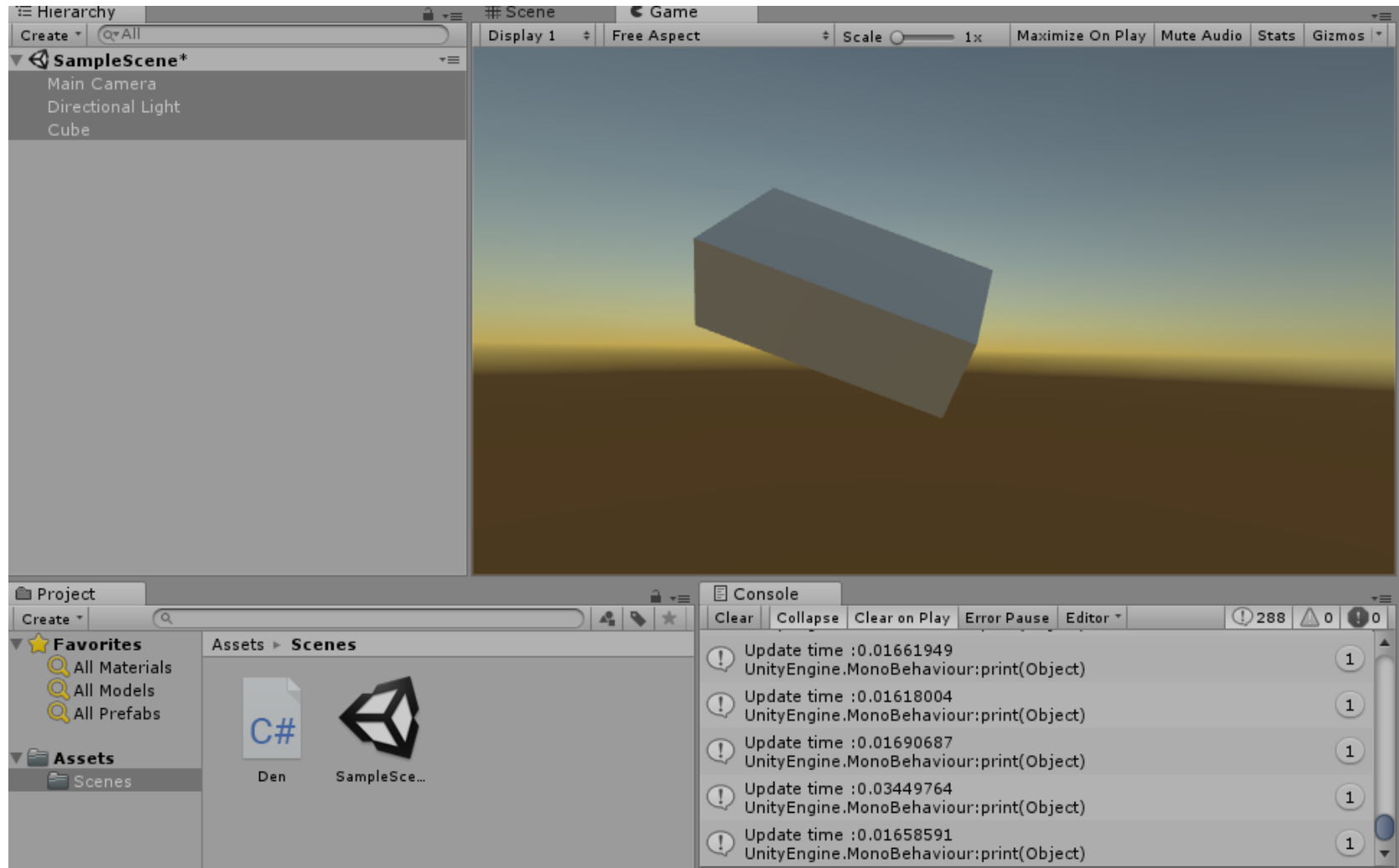
public class Den : MonoBehaviour {

    void Update() {
        print("Update time :" + Time.deltaTime);
    }

    void FixedUpdate() {
        print("FixedUpdate time :" + Time.deltaTime);
    }
}
```

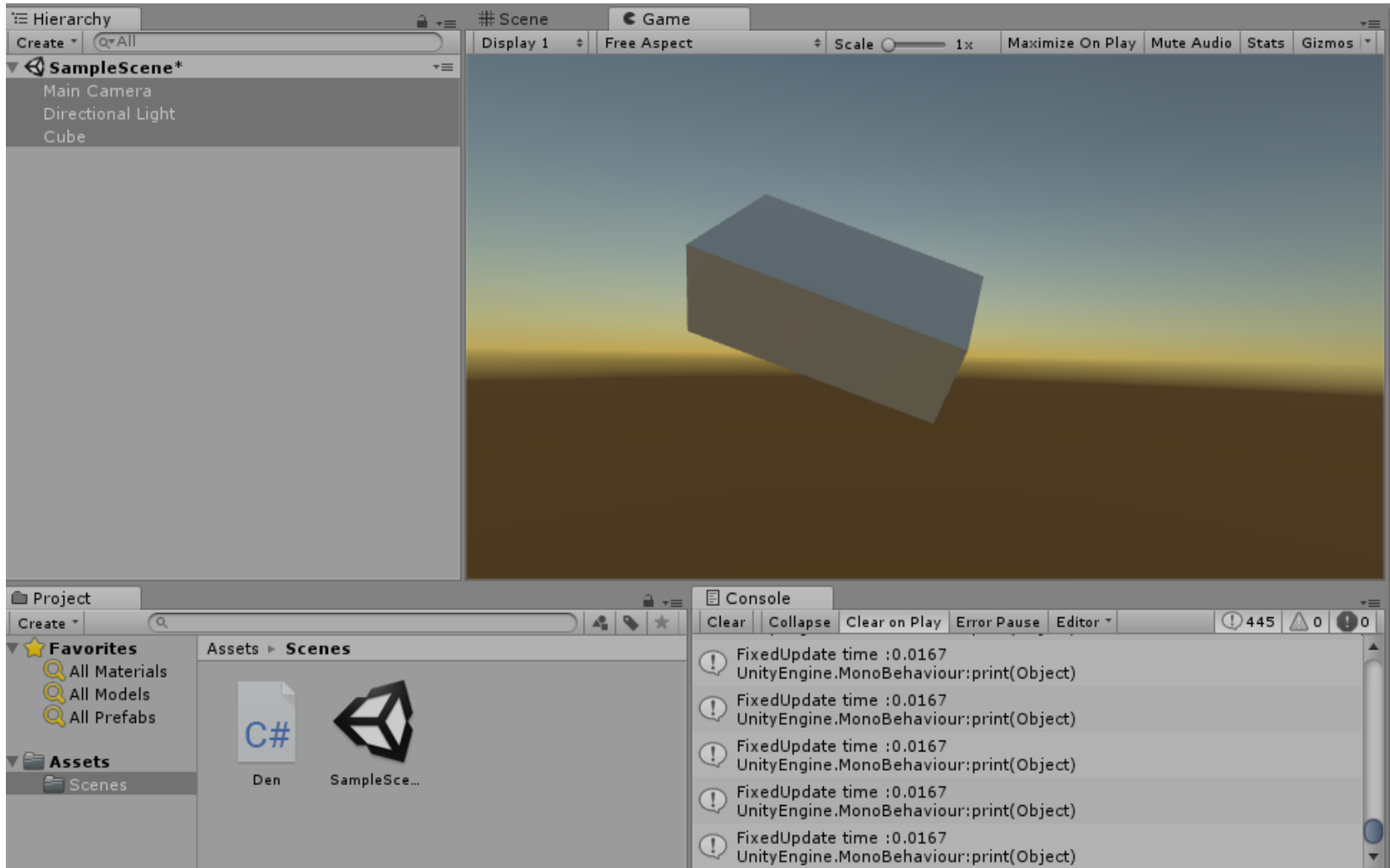


Update vs FixedUpdate





Update vs FixedUpdate





Nhận đầu vào từ bàn phím

```
void Update() {
    if (Input.GetKey(KeyCode.W)) {
        print("Up");
        transform.Translate(0, speed * Time.deltaTime, 0);
    }
    if (Input.GetKey(KeyCode.S)) {
        print("Down");
        transform.Translate(0, -speed * Time.deltaTime, 0);
    }
    if (Input.GetKey(KeyCode.A)) {
        print("Left");
        transform.Translate(-speed * Time.deltaTime, 0, 0);
    }
    if (Input.GetKey(KeyCode.D)) {
        print("Right");
        transform.Translate(speed * Time.deltaTime, 0, 0);
    }
}
```