**Mục Lục (Table of Contents)**

1. Giới Thiệu về Ngôn Ngữ C# trong Unity 1.1. Tại sao nên chọn C# cho Unity? 1.2. Các tính năng nổi bật của C# \* Ngôn ngữ biên dịch (Compiled Language) \* Mã được quản lý (Managed Code) \* Ngôn ngữ định kiểu mạnh (Strongly Typed Language) \* Lập trình hướng chức năng (Function-Based Programming) \* Lập trình hướng đối tượng (Object-Oriented Programming - OOP)
2. Làm Quen với Script C# trong Unity 2.1. Cách tạo và sử dụng Script 2.2. Cấu trúc cơ bản của một file Script C#
3. Biến (Variables), Kiểu dữ liệu (Data types) và Toán tử (Operators) 3.1. Biến (Variables) trong C# 3.2. Kiểu dữ liệu (Data types) \* Kiểu số (Numbers) \* Kiểu chuỗi (String) \* Kiểu luận lý (Boolean) \* Kiểu cấu trúc (Struct) \* Kiểu liệt kê (Enum) \* Sự bất biến của chuỗi (Immutable String) và StringBuilder 3.3. Ép kiểu (Casting) và Chuyển đổi kiểu (Type Conversions) 3.4. Toán tử (Operators) \* Toán tử số học (Arithmetic Operators) \* Toán tử quan hệ (Relational Operators) \* Toán tử logic (Logical Operators)
4. Các câu lệnh điều khiển luồng (Control Flow Statements) 4.1. Câu lệnh lựa chọn (Selection Statements) \* Câu lệnh if/else \* Câu lệnh switch 4.2. Câu lệnh lặp (Iteration Statements) \* Vòng lặp for \* Vòng lặp foreach \* Vòng lặp while \* Vòng lặp do-while

**1. Giới Thiệu về Ngôn Ngữ C# trong Unity**

**1.1. Tại sao nên chọn C# cho Unity?**

**C#** là ngôn ngữ lập trình được sử dụng rộng rãi và được khuyến nghị nhất cho việc phát triển game trên nền tảng Unity. Dưới đây là những lý do chính:

* **Cộng đồng lớn và hỗ trợ mạnh mẽ (Strong Community and Support):** Có một cộng đồng các nhà phát triển C# rộng lớn và kho tài nguyên phong phú để hỗ trợ bạn trong quá trình học và làm việc.
* **Tiêu chuẩn công nghiệp (Industry Standard):** C# là một ngôn ngữ đa năng, được ứng dụng trong nhiều lĩnh vực như phát triển web và kỹ thuật phần mềm.
* **Tương thích với .NET (Interoperability with .NET):** Ngôn ngữ này tận dụng sức mạnh của .NET framework để cung cấp các tính năng nâng cao.
* **Dễ học (Ease of Learning):** C# tương đối dễ học, đặc biệt là với những người đã có kinh nghiệm lập trình.

**1.2. Các tính năng nổi bật của C#**

* **Ngôn ngữ biên dịch (Compiled Language):** Mã nguồn C# được dịch thành mã máy (machine code) trước khi thực thi , giúp chương trình chạy nhanh và hiệu quả hơn.
* **Mã được quản lý (Managed Code):** C# được thực thi dưới sự kiểm soát của một môi trường runtime (như Common Language Runtime - CLR). Môi trường này tự động xử lý các tác vụ phức tạp như quản lý bộ nhớ (memory management), giảm thiểu rủi ro lỗi rò rỉ bộ nhớ (memory leaks).
* **Ngôn ngữ định kiểu mạnh (Strongly Typed Language):** Mỗi biến (variable) trong C# phải được khai báo với một kiểu dữ liệu cụ thể (specific data type). Điều này giúp phát hiện lỗi sớm ngay từ giai đoạn biên dịch (compilation) và làm cho code dễ đọc, dễ bảo trì hơn.
* **Lập trình hướng chức năng (Function-Based Programming):** C# cho phép chia nhỏ các tác vụ phức tạp thành các hàm (functions) hoặc phương thức (methods) nhỏ hơn, có thể tái sử dụng. Điều này giúp code có tính mô-đun (modularity) và dễ tổ chức.
* **Lập trình hướng đối tượng (Object-Oriented Programming - OOP):** C# là một ngôn ngữ lập trình hướng đối tượng mạnh mẽ, cho phép bạn mô hình hóa các thực thể trong thế giới thực thành các đối tượng (objects). Các khái niệm cơ bản bao gồm:
  + **Lớp (Classes):** Bản thiết kế để tạo ra các đối tượng.
  + **Đối tượng (Objects):** Các thể hiện (instance) của một lớp.
  + **Tính kế thừa (Inheritance):** Tạo các lớp mới dựa trên lớp đã có.
  + **Tính đa hình (Polymorphism):** Khả năng một đối tượng có thể mang nhiều hình thức.
  + **Tính đóng gói (Encapsulation):** Gộp dữ liệu (data) và phương thức (methods) vào trong một lớp, ẩn đi chi tiết cài đặt.

**2. Làm Quen với Script C# trong Unity**

**2.1. Cách tạo và sử dụng Script**

C# là ngôn ngữ scripting chính trong Unity. Bạn có thể tạo script để định nghĩa hành vi của các đối tượng trong game (GameObjects).

**Các bước cơ bản để bắt đầu:**

1. **Tạo một Script C# mới:** Trong cửa sổ **Project**, nhấn chuột phải và chọn **Create > C# Script**.
2. **Viết code:** Mở file script đã tạo bằng trình soạn thảo (ví dụ: Visual Studio) và viết code C# để điều khiển hành vi của GameObject.
3. **Gán Script vào GameObject:** Kéo thả file script vào một GameObject trong cửa sổ **Hierarchy** hoặc cửa sổ **Inspector**.

**Ví dụ thực tế trong Unity:** Giả sử bạn muốn một vật thể (chẳng hạn như một khối lập phương) di chuyển. Bạn có thể tạo một script C# với đoạn code sau và gán nó vào khối lập phương đó:

C#

using UnityEngine;

public class PlayerMovement : MonoBehaviour

{

// Tốc độ di chuyển của đối tượng

public float speed = 5.0f;

void Update()

{

// Lấy dữ liệu từ input của người chơi

float horizontalInput = Input.GetAxis("Horizontal");

float verticalInput = Input.GetAxis("Vertical");

// Tạo vector di chuyển

Vector3 movement = new Vector3(horizontalInput, 0, verticalInput) \* speed \* Time.deltaTime;

// Di chuyển đối tượng

transform.Translate(movement);

}

}

*Ghi chú:*

* **Update():** Phương thức này được gọi một lần mỗi khung hình (frame), rất phù hợp để xử lý các logic game liên tục như di chuyển, kiểm tra input...
* **Input.GetAxis():** Lấy giá trị đầu vào từ bàn phím (mặc định là các phím mũi tên hoặc A, S, D, W).
* **Vector3:** Một kiểu dữ liệu đặc biệt của Unity để lưu trữ vị trí 3D (x, y, z).
* **transform.Translate():** Phương thức để di chuyển GameObject.

**3. Biến (Variables), Kiểu dữ liệu (Data types) và Toán tử (Operators)**

**3.1. Biến (Variables) trong C#**

Một biến (variable) là một vùng nhớ được đặt tên (named location in memory) dùng để lưu trữ dữ liệu. Giống như một cái hộp, nó có thể chứa các loại dữ liệu khác nhau (số, chữ, văn bản...).

**Ví dụ trong Unity:** Khi tạo một game, bạn cần lưu trữ thông tin về nhân vật, vật phẩm, điểm số... Mỗi thông tin đó sẽ được lưu trong một biến.

C#

public class GameManager : MonoBehaviour

{

// Biến để lưu trữ điểm số của người chơi (số nguyên)

public int playerScore = 0;

// Biến để lưu trữ tên của người chơi (chuỗi ký tự)

public string playerName = "Hero";

// Biến để kiểm tra trạng thái game (đúng/sai)

public bool isGameOver = false;

}

**3.2. Kiểu dữ liệu (Data types)**

Mỗi biến trong C# phải có một kiểu dữ liệu cụ thể, để máy tính biết cách xử lý dữ liệu bên trong.

**Các kiểu dữ liệu phổ biến:**

* **Kiểu số (Numbers):** Chia thành hai loại chính:
  + **Kiểu số nguyên (Integer Types):** Dùng để lưu trữ số nguyên (không có phần thập phân). Ví dụ:

int, long.

* + - int: Kiểu số nguyên 32-bit có dấu. Thường được dùng cho các biến đếm, điểm số...
    - long: Kiểu số nguyên 64-bit có dấu. Dùng khi giá trị quá lớn, vượt quá khả năng lưu trữ của

int.

* + **Kiểu số thực (Floating-Point Types):** Dùng để lưu trữ số có phần thập phân. Ví dụ:

float, double.

* + - float: Kiểu số thực 32-bit. Thường được dùng trong Unity cho các phép tính liên quan đến vị trí, vật lý...
    - double: Kiểu số thực 64-bit. Có độ chính xác cao hơn

float.

* **Kiểu chuỗi (String):** Dùng để lưu trữ văn bản, được đặt trong dấu ngoặc kép (" ").
* **Kiểu luận lý (Boolean):** Chỉ có hai giá trị là true hoặc false. Thường dùng để kiểm tra điều kiện.
* **Kiểu cấu trúc (Struct):** Một kiểu dữ liệu do người dùng tự định nghĩa, nhẹ hơn lớp (class). Thường dùng cho các cấu trúc dữ liệu đơn giản.
* **Kiểu liệt kê (Enum - Enumeration):** Gán tên hằng (constant names) cho một nhóm các giá trị số nguyên. Điều này giúp code dễ đọc và dễ bảo trì hơn, thay vì sử dụng các con số "ma thuật" (magic numbers).
  + **Ví dụ:** Thay vì dùng int day = 0 để chỉ ngày thứ Hai, bạn có thể dùng WeekDays.Monday.

**4. Các câu lệnh điều khiển luồng (Control Flow Statements)**

**4.1. Câu lệnh lựa chọn (Selection Statements)**

* **Câu lệnh if/else:** Cho phép chương trình thực thi một khối lệnh cụ thể dựa trên một điều kiện luận lý (Boolean expression).
  + **Ví dụ trong Unity:**

C#

public class HealthSystem : MonoBehaviour

{

public int currentHealth = 100;

void TakeDamage(int damageAmount)

{

currentHealth -= damageAmount;

if (currentHealth <= 0)

{

Debug.Log("Game Over!");

}

else

{

Debug.Log("Player is still alive!");

}

}

}

* **Câu lệnh switch:** Lựa chọn một trong các danh sách câu lệnh để thực thi dựa trên kết quả so sánh một giá trị. Rất hữu ích khi bạn có nhiều trường hợp cần xử lý.
  + **Ví dụ trong Unity:**

C#

public enum PlayerState { Idle, Running, Jumping, Attacking }

public class PlayerController : MonoBehaviour

{

public PlayerState currentState;

void HandleState()

{

switch (currentState)

{

case PlayerState.Running:

// Thực hiện hành động chạy

break;

case PlayerState.Jumping:

// Thực hiện hành động nhảy

break;

case PlayerState.Attacking:

// Thực hiện hành động tấn công

break;

default:

// Hành động mặc định (ví dụ: đứng yên)

break;

}

}

}

**4.2. Câu lệnh lặp (Iteration Statements)**

* **Vòng lặp for:** Lặp lại một khối lệnh một số lần xác định.
* **Vòng lặp foreach:** Lặp qua từng phần tử trong một tập hợp dữ liệu (collection).
* **Vòng lặp while:** Lặp lại một khối lệnh khi một điều kiện cho trước vẫn còn đúng.
* **Vòng lặp do-while:** Tương tự while, nhưng đảm bảo khối lệnh sẽ được thực thi ít nhất một lần trước khi kiểm tra điều kiện.