Lab 3 (4 tiết): Introduction to Scripting

MỤC ĐÍCH

Bài lab này giúp sinh viên tìm hiểu một số nguyên lý trong lập trình Unity bằng C#.

YÊU CẦU

- Sinh viên đọc toàn bộ phần nội dung và thực hiện theo hướng dẫn.

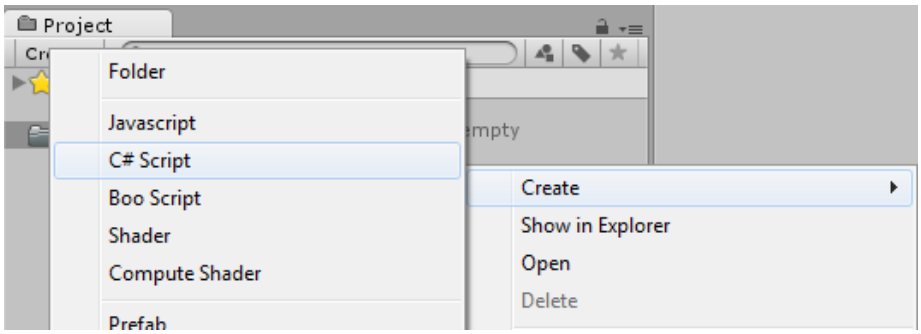
- Sau đó thực hiện các bài tập tương ứng.

NỘI DUNG

1. Thành phần Script trong Unity

Trong Unity, Script tồn tại dưới dạng một Component, nghĩa là bạn phải thêm vào một thành phần là Script.

Tạo một dự án có tên là Lession 3. Vào Project View, tạo một thư mục có tên là Scripts, sau đó click phải, chọn Create, chọn C# Script, đặt tên là "Rotate".



Nhấp đôi chuột lên tập tin vừa tạo, mở bằng Mono Develop hoặc Visual Studio, nội dung trong tập tin đã bao gồm hai phương thức là Start() và Update():

- Phương thức Start() được gọi khi bắt đầu Game.

- Phương thức Update() được gọi khi bắt đầu mỗi Frame.

Lưu ý: *Có 2 phương thức là Update() và FixedUpdate(): Update thực hiện cập nhật tham số sau mỗi frame, còn FixedUpdate thực hiện cập nhật tham số có thể là 0 lần, 1 lần hoặc nhiều lần trong frame tùy thuộc vào các tác động vật lý khi thực hiện bởi người dùng.*

Viết code trong phương thức Update như hình sau (màu đỏ):

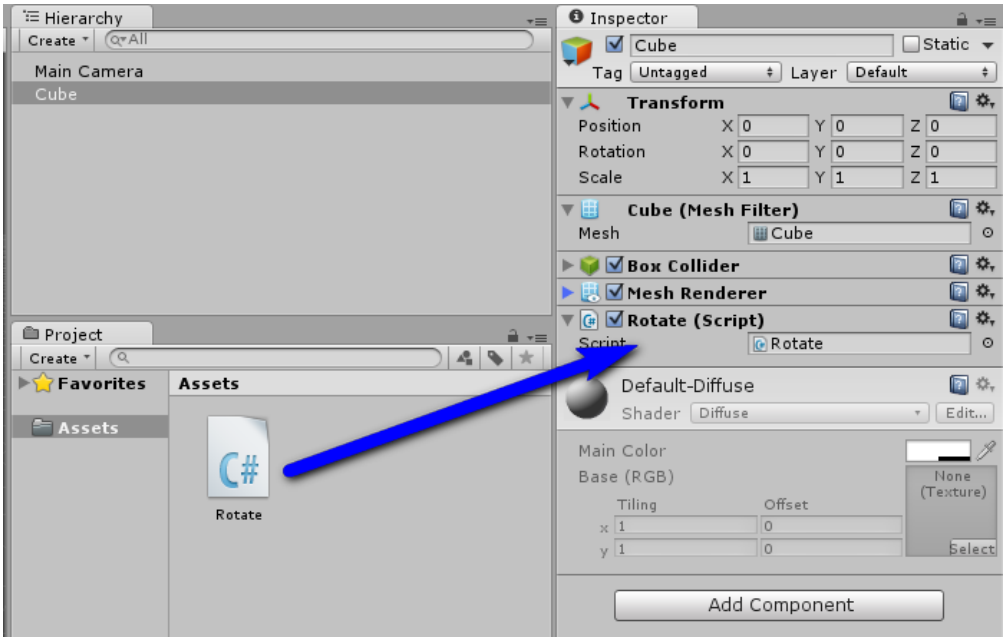
```
// Use this for initialization
void Start () {
}
// Update is called once per frame
void Update () {
    transform.Rotate(2, 0, 0);
}
```

Giải thích:

- transform: là thành phần Transform của Game Object.

- Phương thức Rotate(x,y,z) sẽ quay đối tượng 2 đơn vị theo trục x, y, z mỗi khi 1 frame được gọi. Các tham số trong hàm này nếu bằng 0 sẽ không quay theo trục đó, ví dụ: Rotate(2,0,0) sẽ quay theo trục x.

Trên Scene, tạo một đối tượng Cube, sau đó kéo tập tin Rotate vừa tạo vào Component của Cube.



Chạy Game ta thấy đối tượng quay quanh trục X.

2. Sử dụng biến trên Component

Unity cho phép khai báo biến dạng public, khi đó giá trị sẽ xuất hiện trên Inspector để người dùng có thể thay đổi tham số thiết lập.

Khai báo thêm biến public _speedRotate trong tập tin Rotate như sau:

```
public float _speedRotate = 2.0f;
// Use this for initialization
void Start () {
}
// Update is called once per frame
void Update () {
    transform.Rotate(_speedRotate, 0, 0);
}
```

Khi đó, trên Inspector, sẽ xuất hiện giá trị tham biến như sau:

Lưu ý:

- Có thể thay đổi giá trị của tham biến bằng cách gõ vào hoặc nhấp và kéo chuột.
- Thông thường, để bảo mật biến tốt hơn, Unity cung cấp cơ chế *SerializeFiled*, lúc này có thể khai báo biến dạng *private*, và thêm từ khóa trên đầu như sau:

```
[SerializeField]
float _speedRotate = 2.0f;
```

3. Thay đổi tốc độ không phụ thuộc Frame

Trong trường hợp code trên, tốc độ quay phụ thuộc vào Frame, điều này bị ảnh hưởng bởi máy tính (tốc độ, card đồ họa...). Người ta muốn tính lại tốc độ để không phụ thuộc vào yếu tố phần cứng mà phụ thuộc yếu tố thời gian. Khi đó thay vì nói "*rotate by 2 degrees every frame*", chúng ta sẽ nói là "*rotate 2 degrees every second*".

Khi đó, người ta nhân tốc độ với tham số *Time.deltaTime* như sau:

```
// Update is called once per frame
void Update () {
    transform.Rotate(_speedRotate*Time.deltaTime, 0, 0);
}
```

Time.deltaTime là tham số cho biết thời gian để thực hiện một frame.

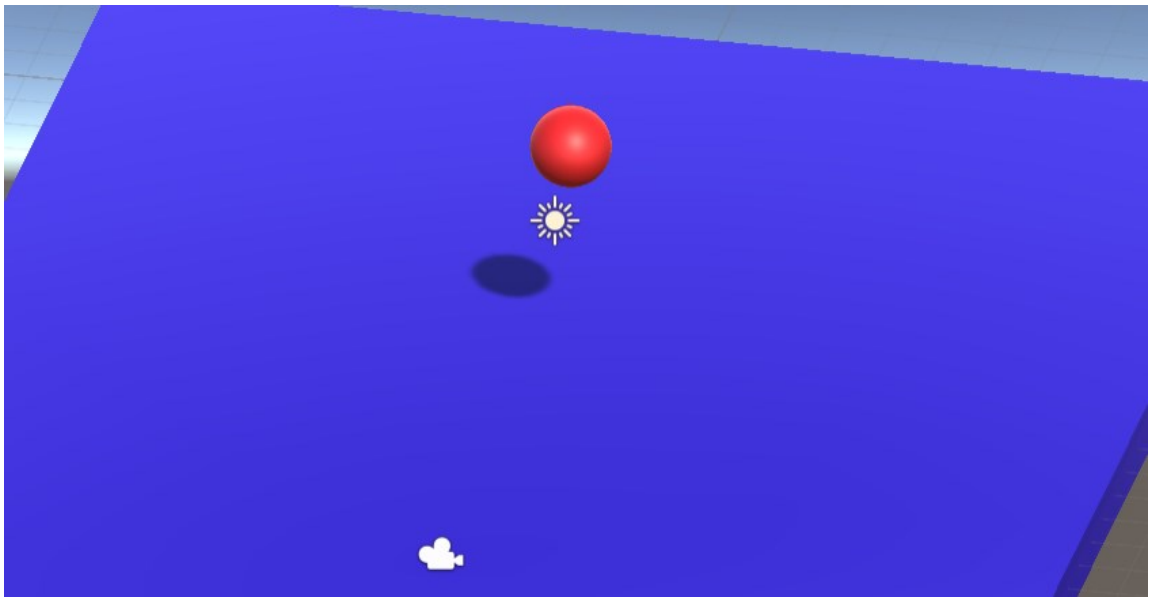
Lưu lại và chạy Game, sẽ thấy tốc độ bị giảm xuống, có thể thay đổi giá trị để tăng tốc độ lên. Để thực hiện chuyển động một cách hiệu quả, sinh viên nên lấy tốc độ nhân với *Time.deltaTime* để đưa về trạng thái không phụ thuộc phần cứng cho game.

4. Di chuyển khối hộp

Sử dụng Cube, tạo một mặt phẳng, thêm vào một khối cầu nằm trên mặt phẳng (có thể gắn RigidBody và Collider để khối cầu rơi xuống và nằm trên mặt phẳng), thực hiện tạo Script tương tự như trên, nhưng có thể điều khiển khối hộp di chuyển lên phía trước và qua trái, phải bằng lệnh *Translate* như sau trong hàm Update():

```
public class RotateSphare : MonoBehaviour
{
    [SerializeField]
    private float _speed = 2.0f;
    // Update is called once per frame
    Unity Message | 0 references
    void Update()
    {
        float x = Input.GetAxis("Horizontal") * _speed * Time.deltaTime;
        float z = Input.GetAxis("Vertical") * _speed * Time.deltaTime;
        transform.Translate(x,0,z);
    }
}
```

Trong đó từ khóa *Vertical* hoặc *Horizontal* tương ứng là phím mũi tên dọc (lên, xuống) và ngang (trái, phải). Hãy thực hiện thêm lệnh để khối hộp có thể di chuyển lên, xuống và qua trái, phải.



Dùng lệnh Input.GetKeyUp(KeyCode….) để dùng một số phím điều khiển khối hộp di chuyển, quay, co dãn theo một giá trị nhất định.

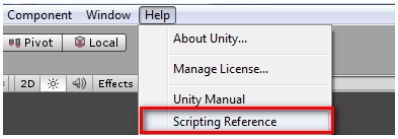
Một giải pháp khác là sinh viên sử dụng hàm AddForce của RigidBody, chi tiết xem tại đây:

https://www.youtube.com/watch?v=Q52bJCfUXfg

5. Một số trợ giúp về Script

5.1. Official Documentation

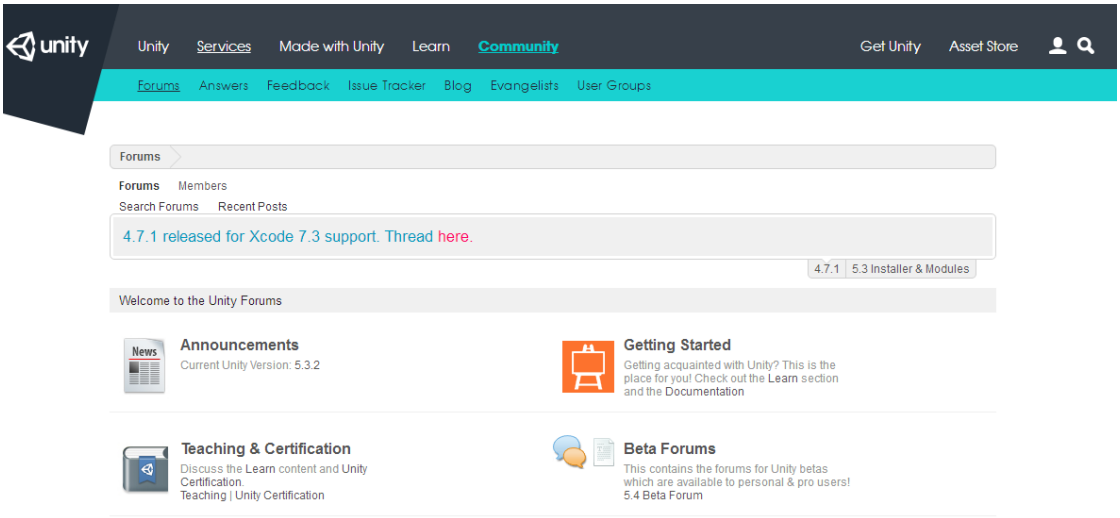
Unity cung cấp các Script hướng dẫn offline băng cách vào Help > Scripting Reference



5.2. Community Forums

Sinh viên có thể tham gia đặt câu hỏi và xem các câu trả lời, giải đáp tại:

http://forum.unity.com/.



BÀI TẬP

Bài 1.

Trên ví dụ về khối hộp, thêm vào các đối tượng khác như Capsule, Cylinder… rồi thêm màu sắc, đèn chiếu sáng và kéo Script vào các đối tượng này để thấy phép quay ảnh hưởng lên đối tượng. Có thể thay đổi các tham số lên các trục và điều chỉnh các tham số để thấy phép quay nhanh hay chậm.

Bài 2.

Thực hiện như sau để giúp 1 Capsule khi nhấn phím Space Bar (phím cách) thì nhảy lên 1 khoảng, và sau đó rơi xuống.

- Đưa vào 1 Plane và một Capsule. Reset để Plane ở vị trí (0,0,0), Capsule ở vị trí (0,1,0). Lúc này Capsule sẽ nằm trên bề mặt của Plane.

- Thêm *Rigid Body* cho Capsule, để khi Capsule nhảy lên thì tự rơi xuống.

- Tạo một Script tên là Jumping, gắn vào cho Capsule, code được viết như sau:

```
// Update is called once per frame
void Update () {
        if (IsGround()) // Nếu là mặt đất
        {
            if (Input.GetKeyUp(KeyCode.Space)) // Nhấn nút Space
            {
                // Dịch lên trên 1 đoạn theo chiều y
              transform.Translate(0, _jumpSpeed * Time.deltaTime, 0);
            }
        }
}
private bool IsGround()
{
    // Nếu vị trí y <= 1 là mặt đất
    return (transform.position.y <= 1);
}
```

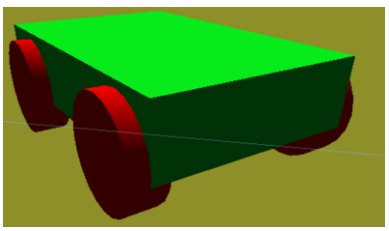
Sinh viên chỉnh sửa thêm để Capsule có thể vừa di chuyển (khi nhấn các phím mũi tên), vừa nhảy lên khi nhấn phím cách.

Bài 3. Tạo game Roll a ball như các bài học trong link sau đây:

https://www.youtube.com/watch?v=zlnJP0vloDg&list=PL-ptF2slHtJAYSWWJ8aqbf1a5tu9u7pAb

Bài 4. Thiết kế 1 xe ô tô đơn giản như trong Hình sau, khi nhấn phím mũi tên thì xe có thể di chuyển và các bánh sẽ xoay theo.

* Đặc biệt: sinh viên cải tiến để khi nhấn phím mũi tên trái, phải thì 2 bánh trước sẽ xoay theo chiều mũi tên và xe sẽ tiến lên theo hướng đó (mô tả lại tay lái của xe ô tô).



Bài 5:

Sinh viên xây dựng chương trình Chiếc nón kỳ diệu như hình sau, khi người chơi nhấn phím Space thì nón bắt đầu quay.

Code gợi ý:

```
float t; // Biến t biểu diễn tham số tốc độ quay.
void Update () {

    if (Input.GetKeyUp(KeyCode.Space))// Nếu nhấn phím Space
        t = Random.value * 100; // sẽ random 1 tốc độ ban đầu

    if (t > 0)
        t -= 0.01f; // Tốc độ giảm dần về 0
    transform.Rotate(0, t*Time.deltaTime, 0); // Quay chiếc nón
}
```

Bài 6*: Sinh viên nghiên cứu xây dựng trò chơi vòng quay **Sun Wheel**