Lập trình Game
Bài tập Thực hành 02

Vũ Lê Thế Anh (1612838)

1 Vật lý và Điều khiển nhân vật

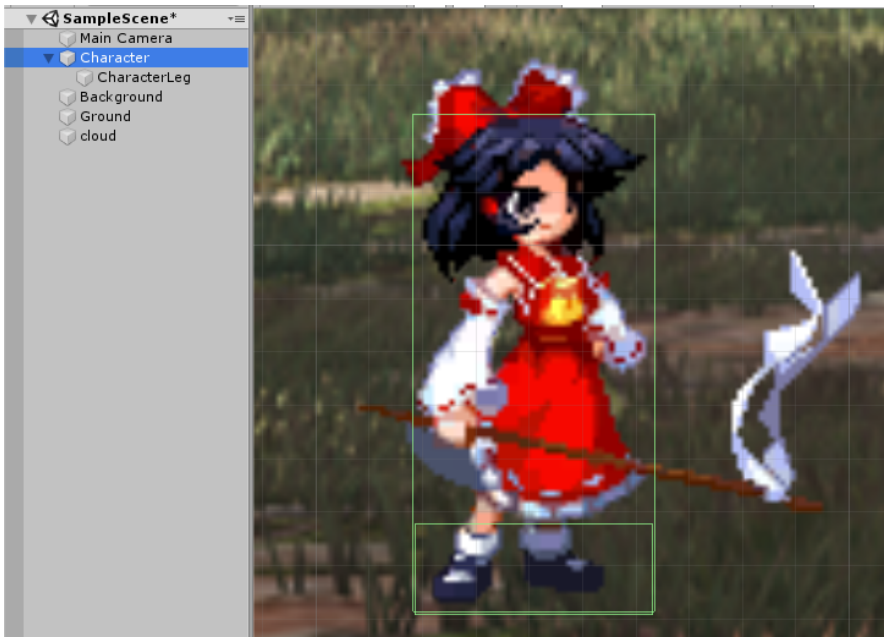
Việc đơn giản có thể nghĩ ra ngay lập tức đó là đặt một biến đếm `jumpCount`, nó sẽ quản lý số lần đã nhảy của nhân vật từ lần chạm đất gần nhất. Mỗi khi người chơi bấm phím nhảy, trò chơi sẽ kiểm tra biến đếm này đã đạt giới hạn cho phép chưa (trong bài tập này là 2 lần), nếu đã vượt thì không cho thực hiện động tác nhảy. Mặt khác, khi nhân vật chạm đất thì biến nhảy này sẽ quay lại bằng 0.

```
1   int jumpCount = 0;
2   int maxJumpTimes = 2;
3
4   void Update() {
5       // [...]
6       if (Input.GetKeyDown(moveUp)) {
7           if (jumpCount < maxJumpTimes) {
8               // Jump here
9               jumpCount += 1;
10          }
11      }
12  }
13
14  private bool IsCollidedGround(Collision2D collision) {
15      Collider2D otherCollider = collision.contacts[0].collider;
16      return otherCollider.tag == "Ground";
17  }
18
19  private void OnCollisionEnter2D(Collision2D collision) {
20      if (IsCollidedGround(collision)) { jumpCount = 0; }
21  }
```

Việc kiểm tra chạm đất lúc này chỉ đơn giản là: thêm tag "Ground" cho mọi vật thể được tính là đất và kiểm tra khi có va chạm thì có đang va chạm vật có tag "Ground" hay không.

Một hạn chế của cách làm này nằm ở việc sự va chạm được tính ở mọi nơi trên người nhân vật, tức là nó có thể chạm bằng đầu thì biến `jumpCount` cũng bị đặt lại. Do đó ta nên giới hạn về việc va chạm chỉ kiểm tra ở chân. Có nhiều cách để làm việc này, người viết chọn cách đơn giản nhất là tạo một `GameObject` con chỉ có Component `BoxCollider2D` ở vùng chân nhân vật.

Khi đó hàm kiểm tra sẽ sửa lại thành



Hình 1: Ví dụ BoxCollider2D của toàn bộ và chân nhân vật.

```
    private bool IsCollidedGround(Collision2D collision) {
        Collider2D myCollider = collision.contacts[0].otherCollider;
        Collider2D otherCollider = collision.contacts[0].collider;
        if (otherCollider.tag == "Ground" &&
            myCollider.name == "CharacterLeg") { jumpCount = 0; }
    }
```

Như vậy người chơi chỉ được nhảy thêm khi nhân vật chạm đất (`otherCollider.tag == "Ground"`) ở phần chân (`myCollider.name == "CharacterLeg"`).

2 Camera 2D

Việc đơn giản nhất là tọa độ (x, y) của Camera luôn được cập nhật bằng tọa độ (x, y) của nhân vật.

```
    public Transform followObj;
    void Update() {
        Vector3 desiredPosition = followObj.transform.position;
        desiredPosition.z = transform.position.z;
        transform.position = smoothedPosition;
    }
```

Việc đặt lại tọa độ z nhằm đảm bảo camera ở vị trí cố định cao hơn (và nhìn xuống) mặt phẳng (plane) 2D của trò chơi (nếu không sẽ không thấy trò chơi).

Một vấn đề với cách làm này là cảm giác thô khi di chuyển nhân vật (nhân vật luôn ở trọng tâm, nền di chuyển chứ nhân vật không di chuyển). Một cách giải quyết là làm cho camera

cập nhật chậm hơn nhân vật. Unity cung cấp một giải pháp cho việc này là thay vì thực hiện trong `Update` ta thực hiện trong `FixedUpdate` (ngoài ra còn có `LateUpdate` nhưng khuyến cáo không dù vì giật, không rõ tại sao), vì hàm này được gọi sau.

Một thay đổi khác sẽ giúp tạo cảm giác mượt hơn là thay vì cập nhật ngay lập tức ta cập nhật theo kiểu nội suy (tạo ra một chuỗi chuyển đổi mượt). Unity cung cấp một giải pháp cho việc này bằng hàm `Lerp` (linear interpolation). Hàm này nhận 2 điểm A, B và một hệ số "tốc độ" chuyển đổi và cho biết điểm phù hợp nằm trên đoạn đường từ A đến B.

```
1     public Transform followObj;
2     public float smoothSpeed = 0.125f;
3     void FixedUpdate() {
4         Vector3 desiredPosition = followObj.transform.position;
5         Vector3 smoothedPosition = Vector3.Lerp(transform.position,
6                                                 desiredPosition,
7                                                 smoothSpeed);
8         smoothedPosition.z = transform.position.z;
9         transform.position = smoothedPosition;
10    }
```

Cách làm này lấy trực tiếp (và y hệt) từ đây.