Bài 4: Sử dụng Sculpting, Texturing và Navigating Terrain

Giảng viên:

Mục tiêu

- Ngôn ngữ kịch bản trong Unity
- Tạo Script trong Assets
- Cấu trúc mặc định của Script
- Attach Scripts vào Objects trong môi trường
- Kết nối với đối tượng trong môi trường, sử dụng ‘tag’
- Sử dụng trình Debug Utility
- Thiết lập và kết nối Script

Ngôn ngữ hỗ trợ trong Unity

- Javascript
- C# (sharp)
- Boo

- Unity documentation cung cấp ví dụ cho từng trường hợp cụ thể.
- Javascript được sử dụng rộng rãi trong các ví dụ, nhưng C# bây giờ trở thành sự lựa chọn đầu tiên trong các Unity tutorials .

Tạo Script Assets

- Từ menu Assets ta chọn Create
- Chọn ngôn ngữ bạn muốn từ menu list
- Tên mặc định của script là NewBehaviorScript
- Đổi tên script thành tên chức năng phù hợp

Ví dụ Javascript

```
#pragma strict

function Start () {

}

function Update () {

}
```

Ví dụ C#

```
using UnityEngine;
using System.Collections;

public class NewBehaviourScript : MonoBehaviour {

	// Use this for initialization
	void Start () {

	}

	// Update is called once per frame
	void Update () {

	}
}
```

File (.js) khi khởi tạo

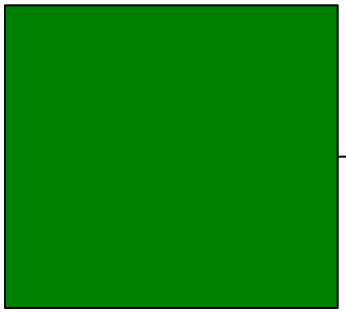
```
#pragma strict

function Start () {
var anIntegerVariable:int = 5;
var afloatVariable:float = 5.0f
}

function Update () {

}
```

Attach Scripts vào Objects

- Tạo một script mới
- Kéo script từ cửa sổ Assets vào object mà bạn muốn.
- Hoặc chọn Add Component từ cửa sổ Inspector và navigate vào panel Scripts.
- Hành vi trong Script có thể hoán đổi thông qua Component checkbox.

Thay đổi thuộc tính của đối tượng thông qua Scripts

Cube Object



ColourSwitcher.js

```
// fragment
function Update () {
if(Input.GetKeyDown(KeyCode.R
)){
gameObject.renderer.material.
color = Color.red;
}
```

Tham chiếu đối tượng bằng cách sử dụng ‘Tags’

- **Tags** – tags là một keywords đơn giản nhằm mục đích gắn đối tượng vào nó trong môi trường. Có thể Tag objects từ trong scripts.
- Tags được dùng tương tự như “instance names” trong ActionScript.
- Thêm tags vào objects is thông qua 2 bước.
- Đầu tiên thêm tag name mà bạn chọn vào list **Tag Manager**.
- Sau đó “apply” nó vào đối tượng bạn chọn.
- Bạn cũng có thể tham chiếu trực tiếp tag name trong scripts.

Thêm một Tags vào đối tượng trong môi trường

e.g. Environment object Sphere with tag 'aSphere'

```
// code fragment
if(hit.collider.gameObject.tag == "aSphere"){
Debug.Log('I collided with the sphere');
}
```

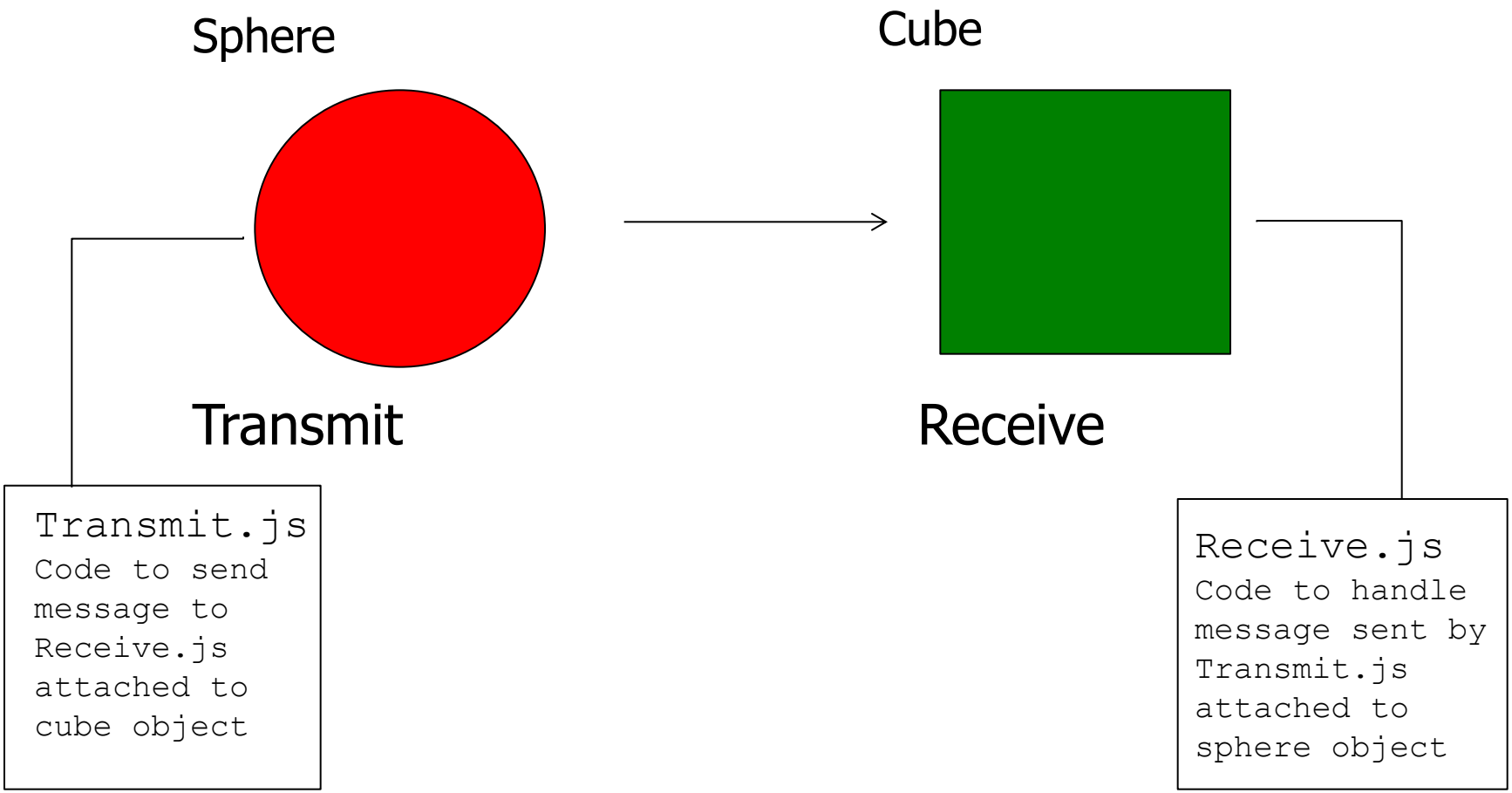
Chức năng Debug Utility

Very useful debug and environment events utility function

```
Debug.Log (“Message you to want to send to the
Console Window”);
```

```
Debug.Log("I have received " + numberOfMessages + ”
messages from the transmit script");
```

Kết nối Inter-Script



Ví dụ Transmit.js Attach vào Sphere Object

```
#pragma strict
// example of script - to - script
communication
// this script transmits messages to the
script attached to the cube

function Start () {
}
function Update () {
 if(Random.value > 0.5f){
  Receive.messageReceived = true;
   }
  else
                ssageReceived = false;
  }
```

Ví dụ Receive.js Attach vào Cube Object

```
#pragma strict
/// this script receive messages from the transmit
script attached to the sphere
static var messageReceived:boolean;// important! static
variable
var numberOfMessages:int;
function Start () {
  messagesReceived = false;
  numberOfMessages = 0;
}
function Update () {
if (messageReceived == true){
    numberOfMessages++;
  Debug.Log("I have received " + numberOfMessages + "
messages from  the transmit script");
  }
else {
     Debug.Log("No messages received");
}
```

Biến Static

- Khai báo biến là **static** làm cho biến luôn luôn có sẵn để các script khác có thể tham chiếu
- Trong lập trình, biến static có phạm vi **global**

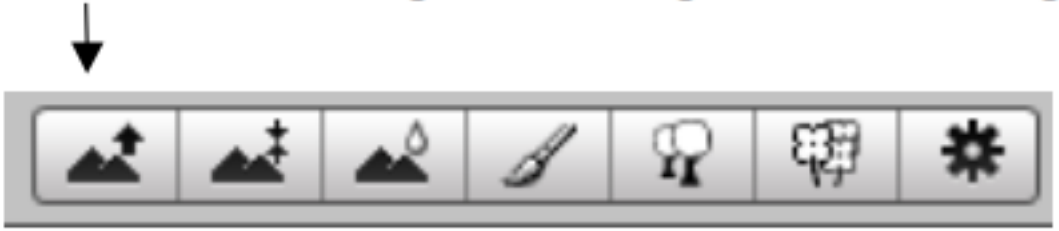
```
// static variable declaration in Receive.js

   static var messageReceived:boolean;
```

Ví dụ: Biến tĩnh Static tham chiếu trong Transmit.js

```
// Transmit.js

 if(Random.value > 0.5f){

  Receive.messageReceived = true;

    }

  else {

  Receive.messageReceived = false;

  }
}
```

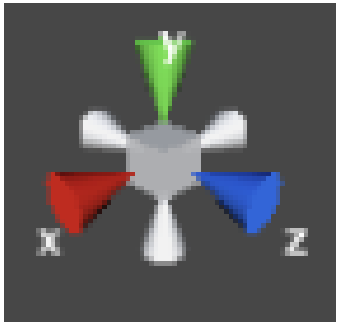
Phương pháp tạo địa hình sử dụng Unity Model Tool

- 1. Tạo một project Unity3D với tên là **TerrainWorld**
- 2. Từ **Screen Menu** ta chọn **Terrain – Create**
- 3. Từ menu **Terrain** ta chọn **Set Resolution** và thay đổi **length, width** với giá trị là **1000**. Để tăng tính trực quan, các địa hình tạo ra sẽ sử dụng nguồn sáng từ screen menu **GameObject – Create Other – Directional Light.** Sử dụng vòng xoay công cụ để chuyển đổi, điều chỉnh ánh sáng ở các địa hình.
- 4. Trong cửa sổ **Hierarchy**, ta chọn **Terrain**. Địa hình mới thay đổi sẽ xuất hiện ở phần cuối của cửa sổ **Inspector**. Chọn công cụ đầu tiên được sử dụng để làm giảm và nâng cao đ



Phương pháp tạo địa hình sử dụng Unity Model Tool

- 5. Chọn brush đầu tiên và thiết lập kích thước brush là 75 và opacity là 40
- 6. Để giúp bạn nhìn thấy các hiệu ứng trực quan, bạn cần thay đổi các khung nhìn khác nhau bằng cách lựa chọn hệ toạ độ theo các trục.

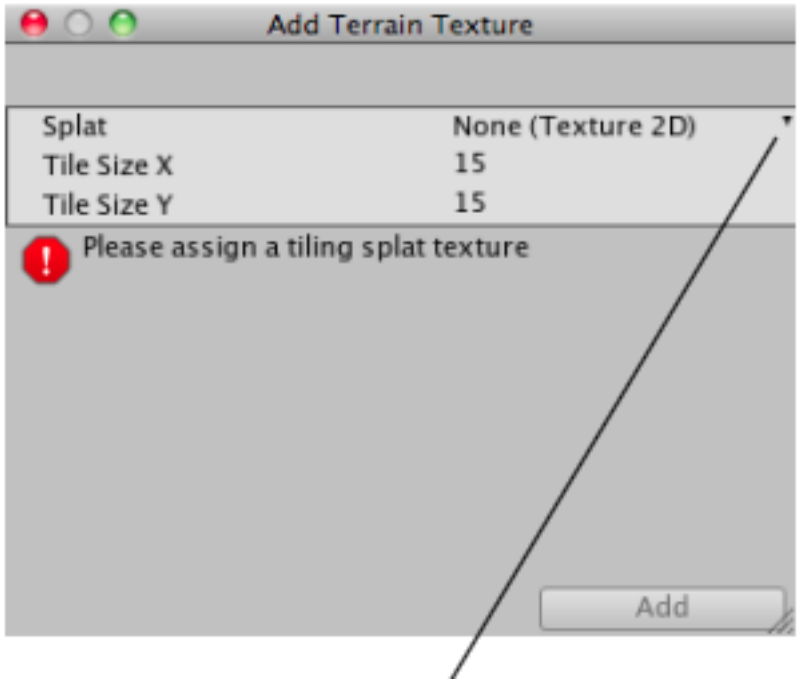


Phương pháp tạo địa hình sử dụng Unity Model Tool

- 7. Bây giờ ta bắt đầu vẽ với brush qua các khu vực địa hình để tăng chiều cao - bằng cách sử dụng phím shift sẽ làm giảm chiều cao. Cố gắng đạt được một địa hình càng mịn càng tốt.
- 8. Để xây dựng được một cao nguyên hay một khu đấtnổi hoặc thang lên xuống hầm mỏ bạn có thể thiết lập chiều cao của địa hình đến một giá trị đặc biệt. Sau đó chọn công cụ thứ hai từ bên trái trong thanh công cụ và thiết lập chiều cao tối đa đến 80. Bây giờ hãy thử tạo một ngọn đồi, một cao nguyên bằng cách tiếp tục dùng brush để vẽ với chiều cao tối. Bạn có thể cần phải các brush khác nhau.

Phương pháp tạo địa hình sử dụng Unity Model Tool

- 9. Chọn công cụ thứ ba từ bên trái trong thanh công cụ. Công cụ này cung cấp cho việc lựa chọn kết cấu để tô màu lên các địa hình. Các kết cấu đầu tiên sẽ bao gồm toàn bộ địa hình và sau đó sử dụng kết cấu tiếp theo để thêm các chi tiết cụ thể hơn. Chọn tab EditTextures, sau đó chọn Add Te

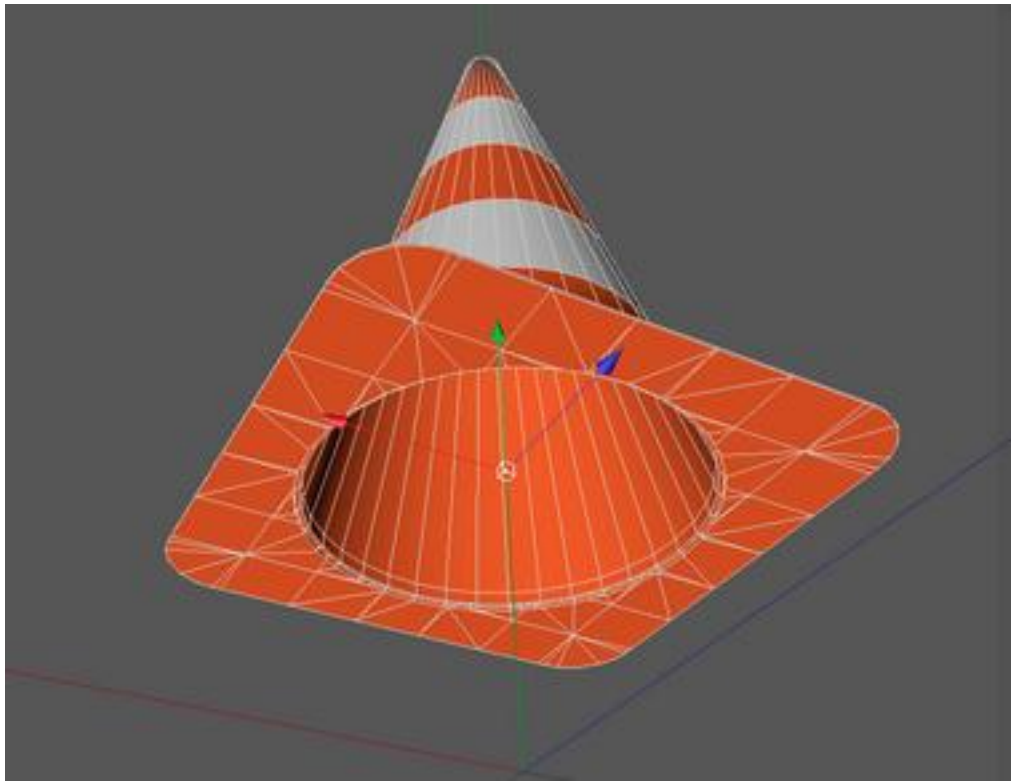


Phương pháp tạo địa hình sử dụng Unity Model Tool

- 10. Bây giờ chọn vào mũi tên nhỏ để xổ ra danh sách các kết cấu có sẵn. Chọn Grass (Hill) sau đó chọn Add. Địa hình nên bây giờ được bao phủ bởi cỏ. Để thêm chi tiết địa phương, bạn cần phải chọn một kết cấu từ danh sách. Ta lặp lại bước 9. chọn Grass & Rock. Sử dụng kết cấu này để thêm màu trời cũng như chi tiết cho các đỉnh và các cạnh của những ngọn đồi và cao nguyên. Hãy thử với brush với các thiết lập khác nhau, lưu ý để opacity thấp.
- 11. Tiếp theo sử dụng một đối tượng prefab để cung cấp character đầu tiên trong môi trường này. Từ cửa sổ Project, ta chọn PreFabs và kéo một character vào cửa sổ Scene. Hãy chắc chắn rằng prefab trên bề mặt của địa hình.
- 12. Chạy script. Sau đó Save scene của bạn là BasicWorld.

Lưới (MESHES)

- Khi một đối tượng mô hình 3D được Import. Unity sẽ gán cho nó một đại diện như một bộ lưới. Lưới phải được đính kèm với một Game Object bằng cách sử dụng một thành phần Filter Meshes.

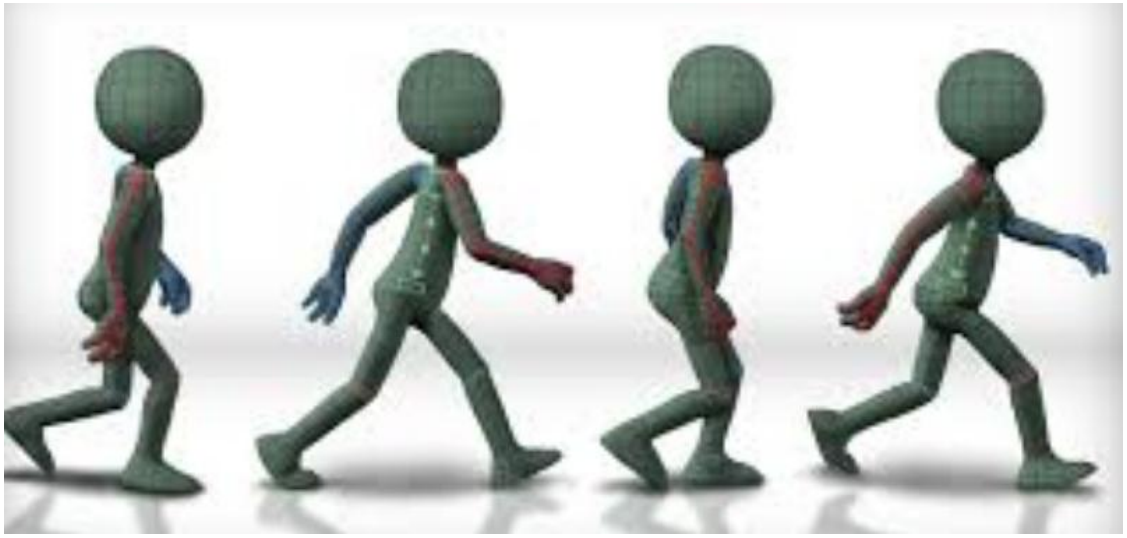


Định dạng model 3D

- Export file định dạng 3D như: *.FBX, *.OBJ.
- Ưu điểm:
  - Chỉ xuất ra dữ liệu bạn cần.
  - Kiểm tra lại dữ liệu.(Import lại)
  - Tập tin khá nhẹ.
  - Khuyến khích phương pháp Module.
  - Hỗ trợ các gói 3D.
- Nhược điểm:
  - Có thể gây chậm trong việc tạo mẫu và sử dụng lại.
  - Khó theo dõi
- Có thể sử dụng các file của 3D MAX, Maya, Blend

Animation View Guide (Legacy)

- Ctrl + 6(Windows): mở Animation View.
- Để sử dụng Animation cơ bản, trước tiên chọn Create new clip. Animation sẽ hiển thị các tùy chọn trên Animation View hoặc trên Inspector.
- Để áp dụng animation cho một đối tượng nào đó bạn phải chọn vào đối tượng đó và thuộc tính animation tương ứng sẽ hiển thị trên Animation View tương ứng.

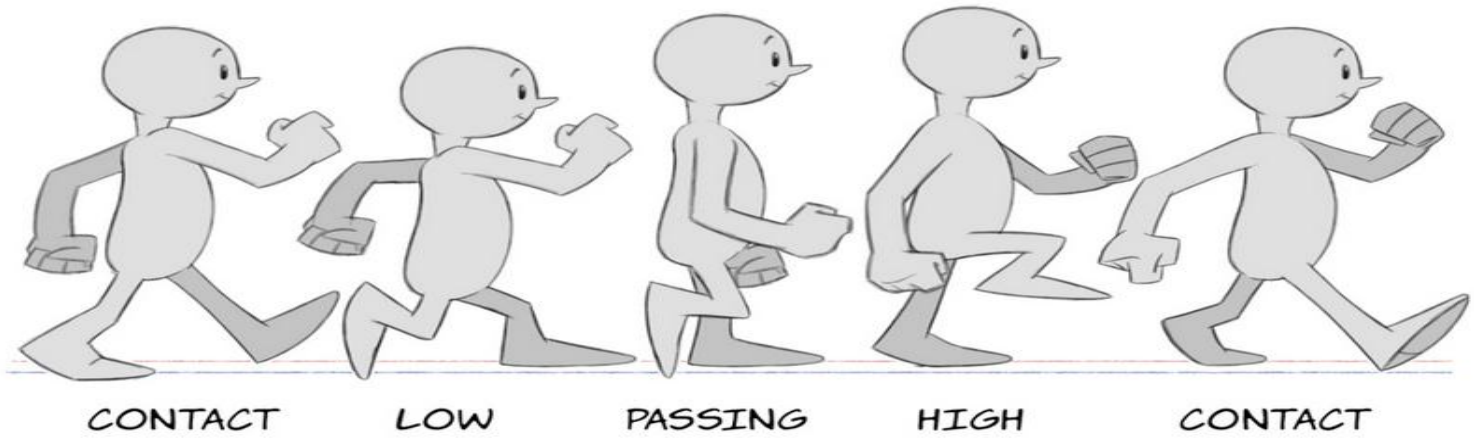


Sử dụng Cuves

- Chọn Right – click và chọn add Curves
- Trên Animation view có thể chọn một thuộc tính hoặc một vào thuộc tính để tùy biến hoạt hình theo các thuộc tính của đối tượng.
- Chỉnh sửa Curves bằng cách tại các frame đã được tạo, kéo đường kẻ theo ý bạn muốn. Nó sẽ hiển thị ngay trên Scene View hoặc trên Inspector
- Tại các điểm Frame, có thể xóa frame, chỉnh/chọn tiếp tuyến
- Có thể sử dụng button nhỏ (+) để thêm Frame hoặc Event cho Animation hoặc click-right lên thanh frame.

Animation Scripting (Legacy)

- Hệ thống Animation của Unity cho phép bạn tạo đẹp nhân vật của mình bằng các pha trộn các vật liệu, chất phụ gia, đồng bộ hóa thời gian, chu kì, các lớp ảnh động, kiểm soát tất cả các các khía cạnh phát lại của hình ảnh động(thời gian, tốc độ, sự pha trộn – trọng lượng)
- Một nhân vật hoạt hình luôn có hai vấn đề chính đó là sự di chuyển của nó trong thế giới và tạo hiệu ứng cho phù hợp.



Animation Blending

- Trong các trò chơi ngày nay, hình ảnh động hòa trộn làm cho nhân vật trở nên trơn chu hơn. Tại bất cứ thời điểm nào trong game, bạn đều có thể chuyển đổi qua lại giữa hình ảnh động và hình ảnh tĩnh
- Đây là nơi các hình ảnh động được Blending. Tất cả các hình ảnh động được thêm, pha trộn vào nhau để được hình ảnh động cuối cùng

Animation Blending

- Ví dụ: Tạo một chuyển động đơn giản để xem nhân vật chuyển động nhanh thế nào, và rồi chuyển đổi giữa hoạt hình động và hoạt hình tĩnh

```
function Update () {
  if (Input.GetAxis("Vertical") > 0.2)
     animation.CrossFade ("walk");
  else
     animation.CrossFade ("idle");
}
```

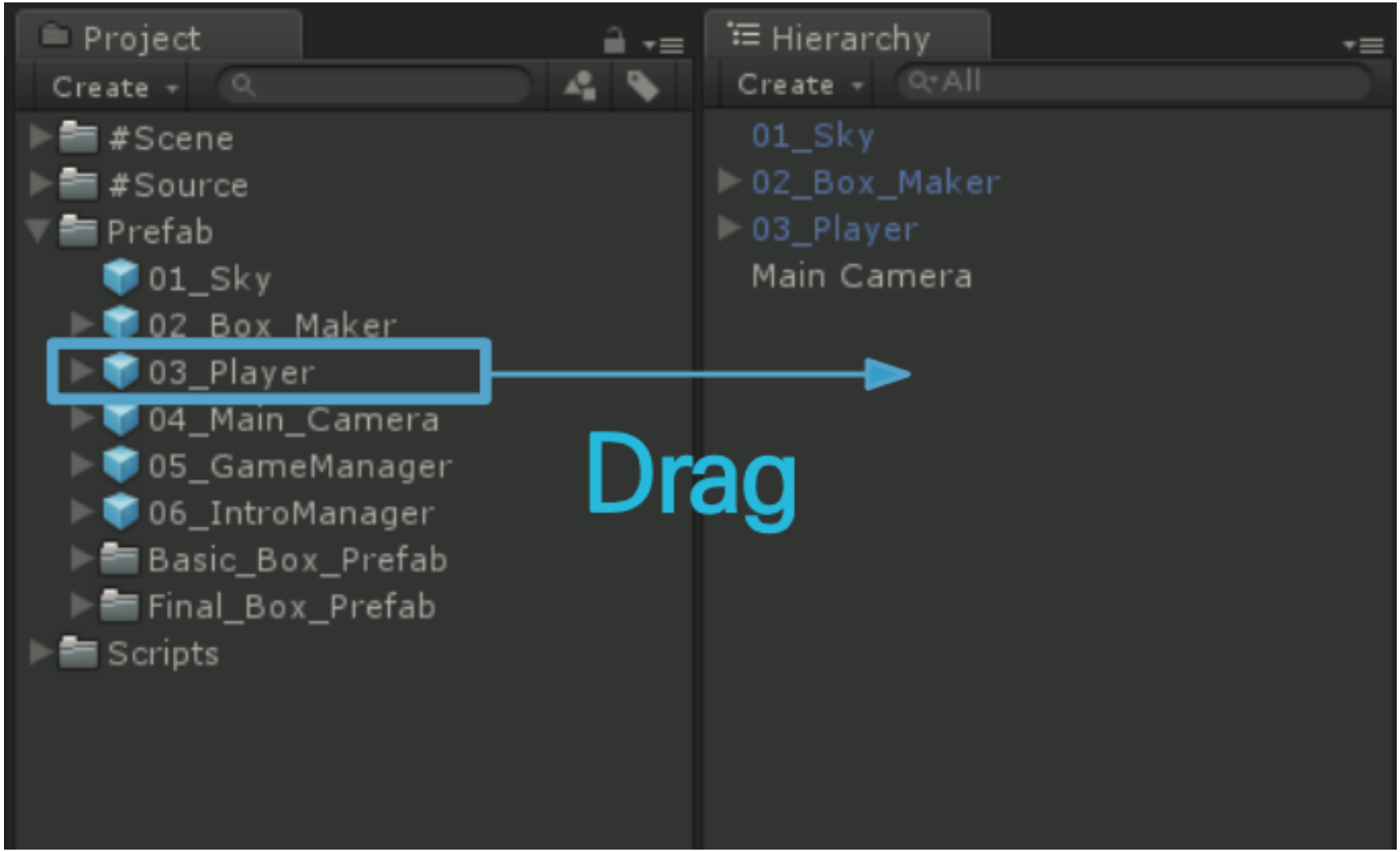
Animation Layers

- Là một khái niệm hữu ích giúp bạn nhóm các hình ảnh động và ưu tiên cho chúng.
- Hệ thống hoạt hình của Unity có thể pha trộn giữa nhiều Clip hoạt hình theo ý bạn muốn. Có thể gán lượng pha trộn bằng tay hay sử dụng animation.CrossFade()
- Lượng Blend luôn được bình thường hóa trước khi được áp dụng. Tưởng tượng như bạn có một chu kỳ Walk và một chu kỳ chạy, cả hai đều có lượng là 1(100%). Khi bình thường hóa thì mỗi chu kỳ chiếm 50% lượng. Tuy nhiên nếu bạn muốn cho bên nào đó có lượng lớn hơn thì bạn có thể trộn bằng tay

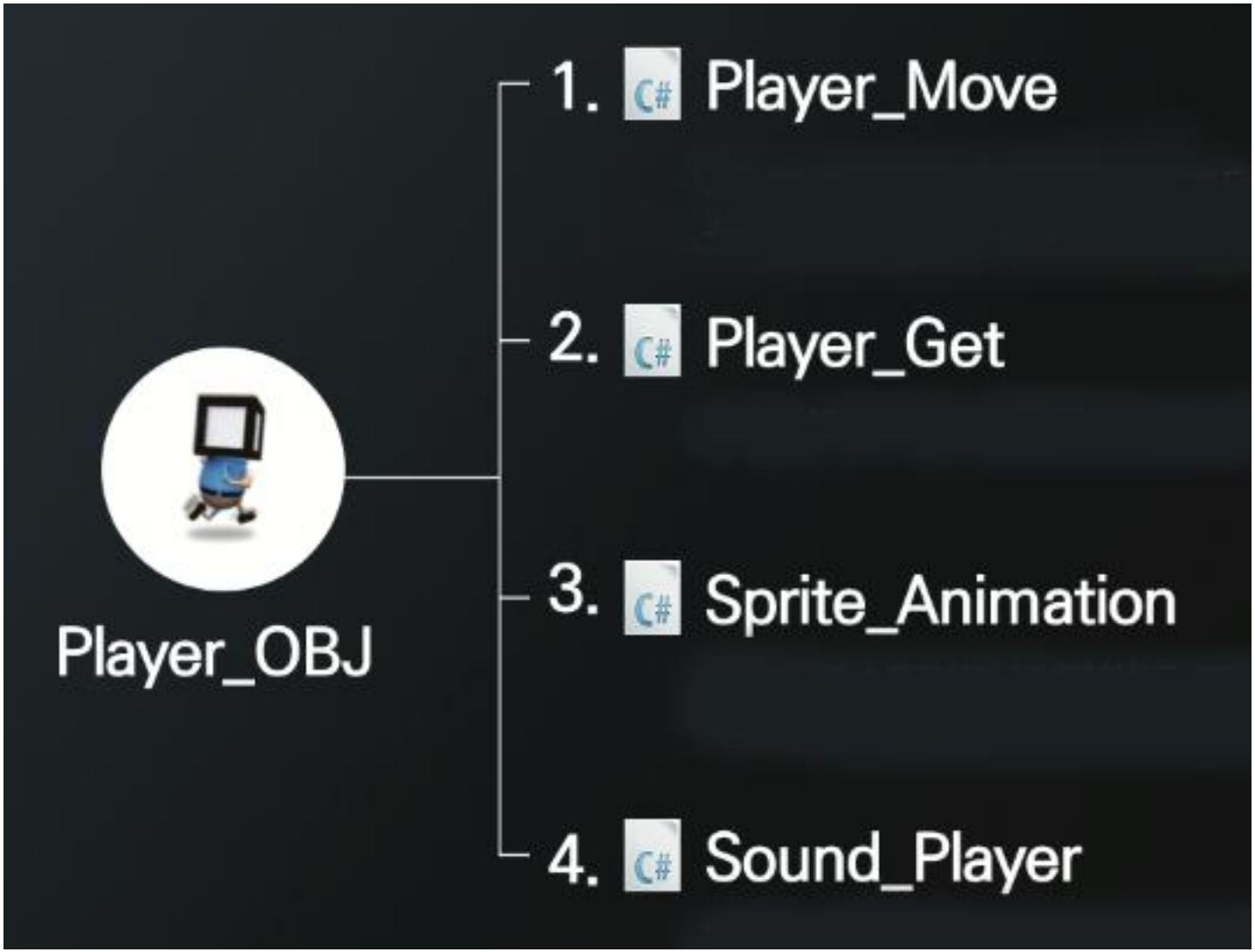
Xây dựng scene Player: (phần 1)

Character chạy trên địa hình và ăn coin

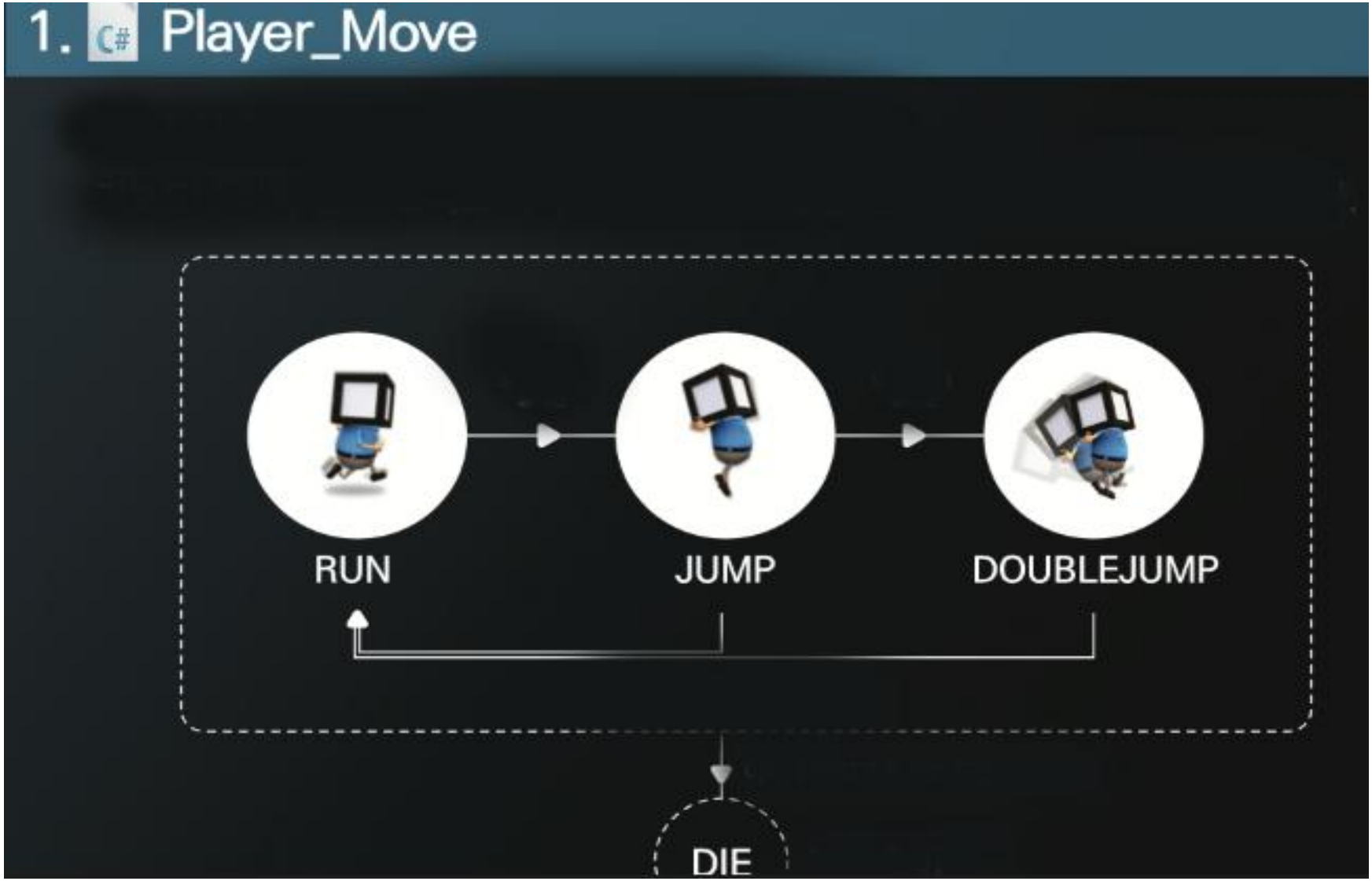
Kéo Prefab vào Hierarchy



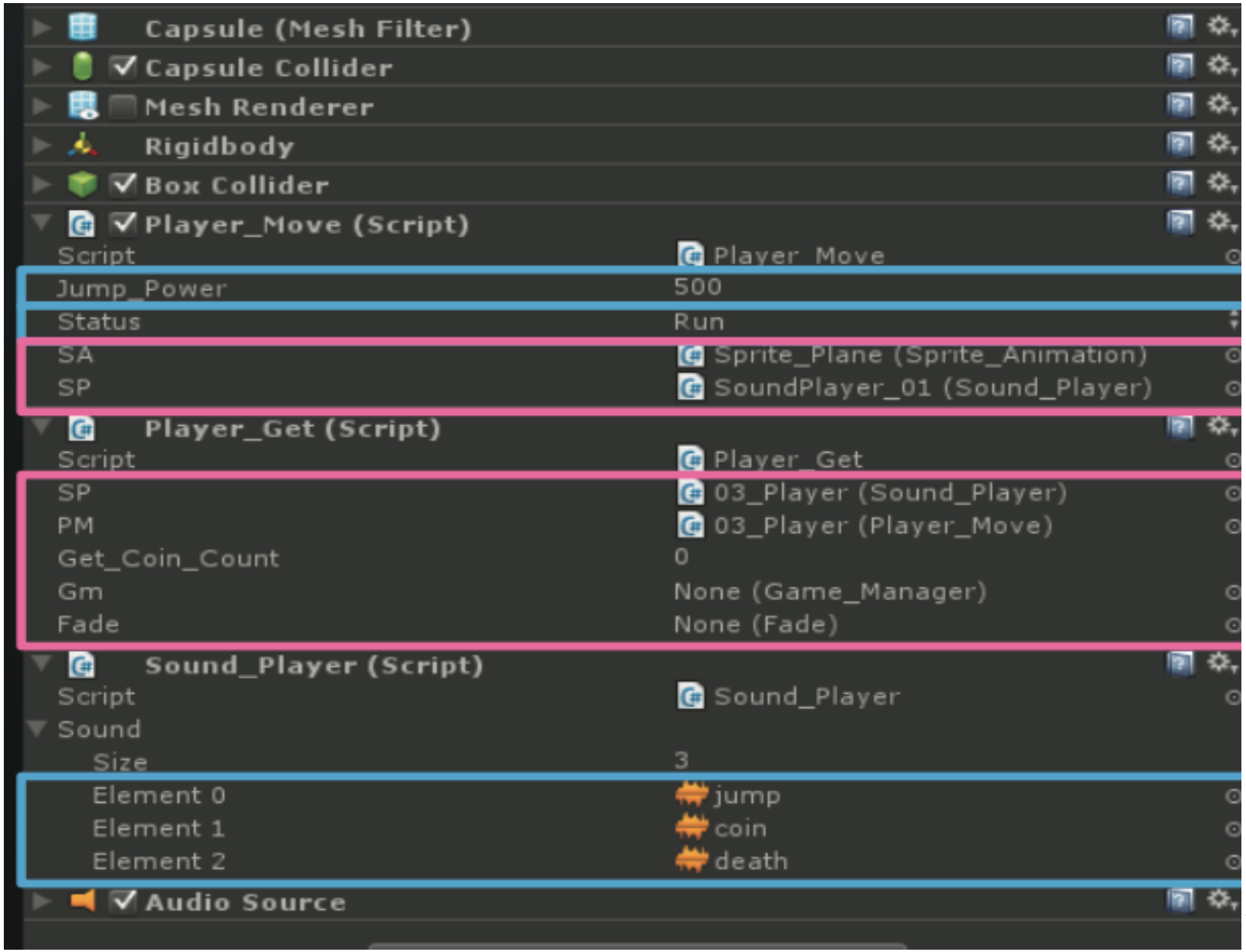
Player được điều khiển bởi 4 phương thức



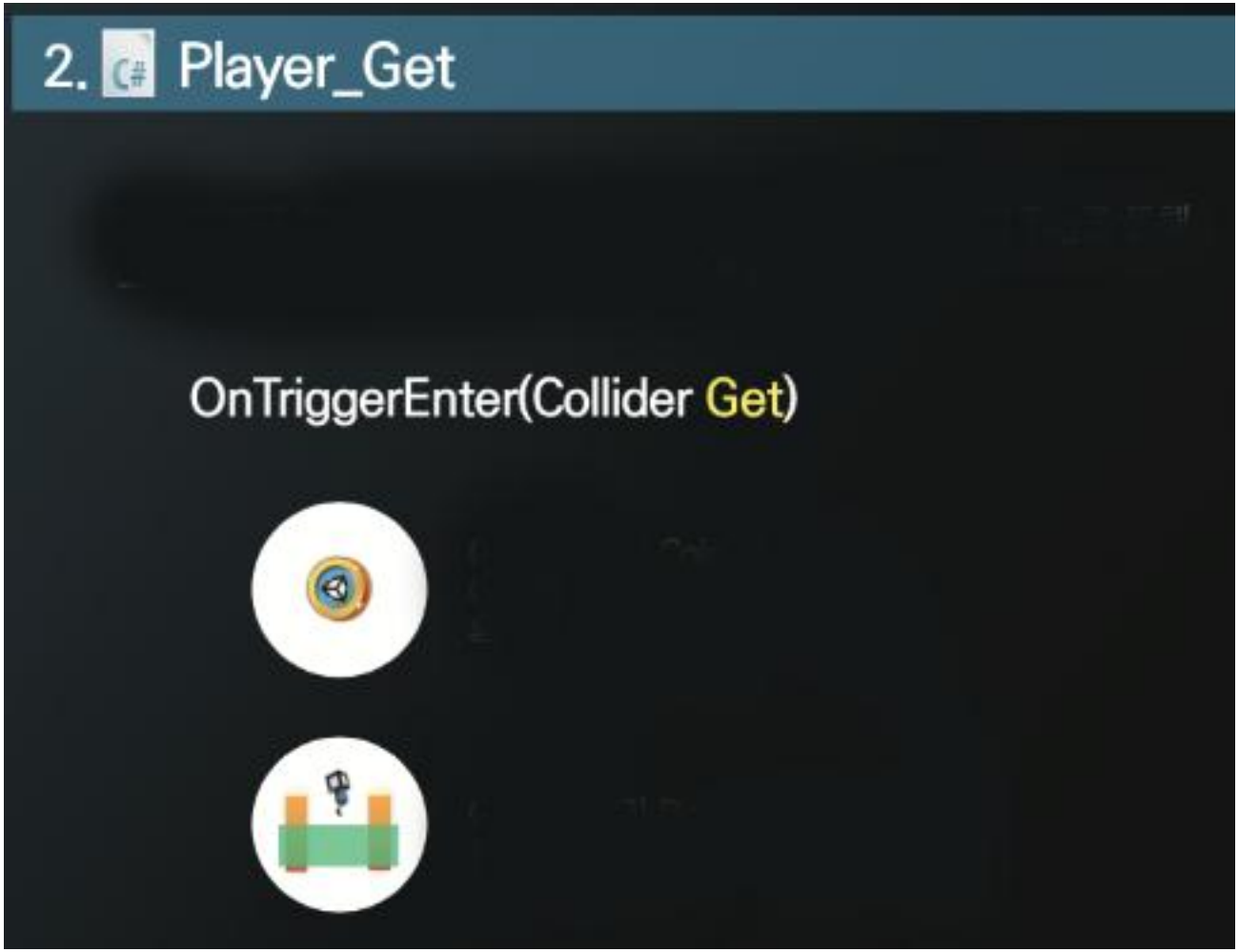
Các trạng thái khi di chuyển



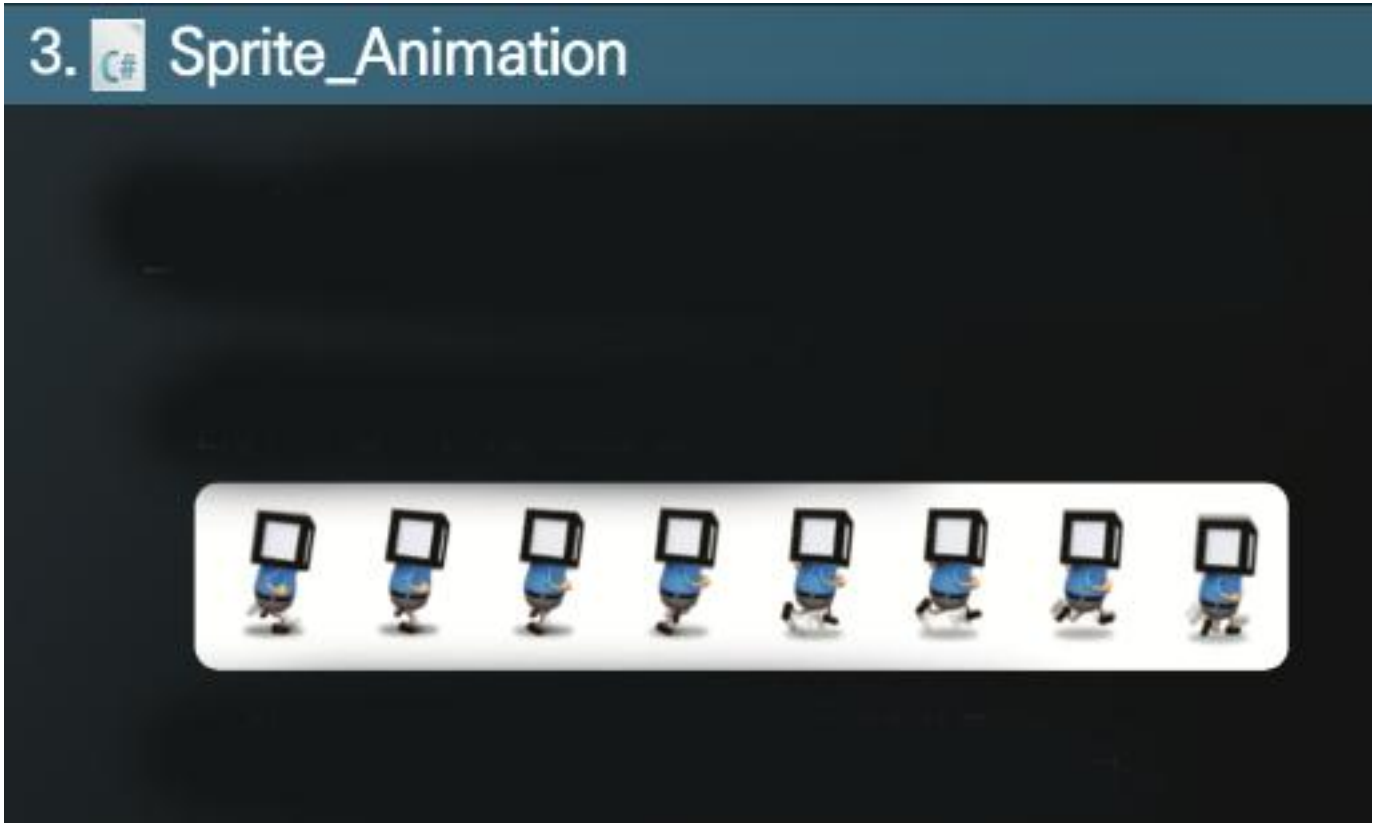
Attach script và thiết lập thông số cho Player



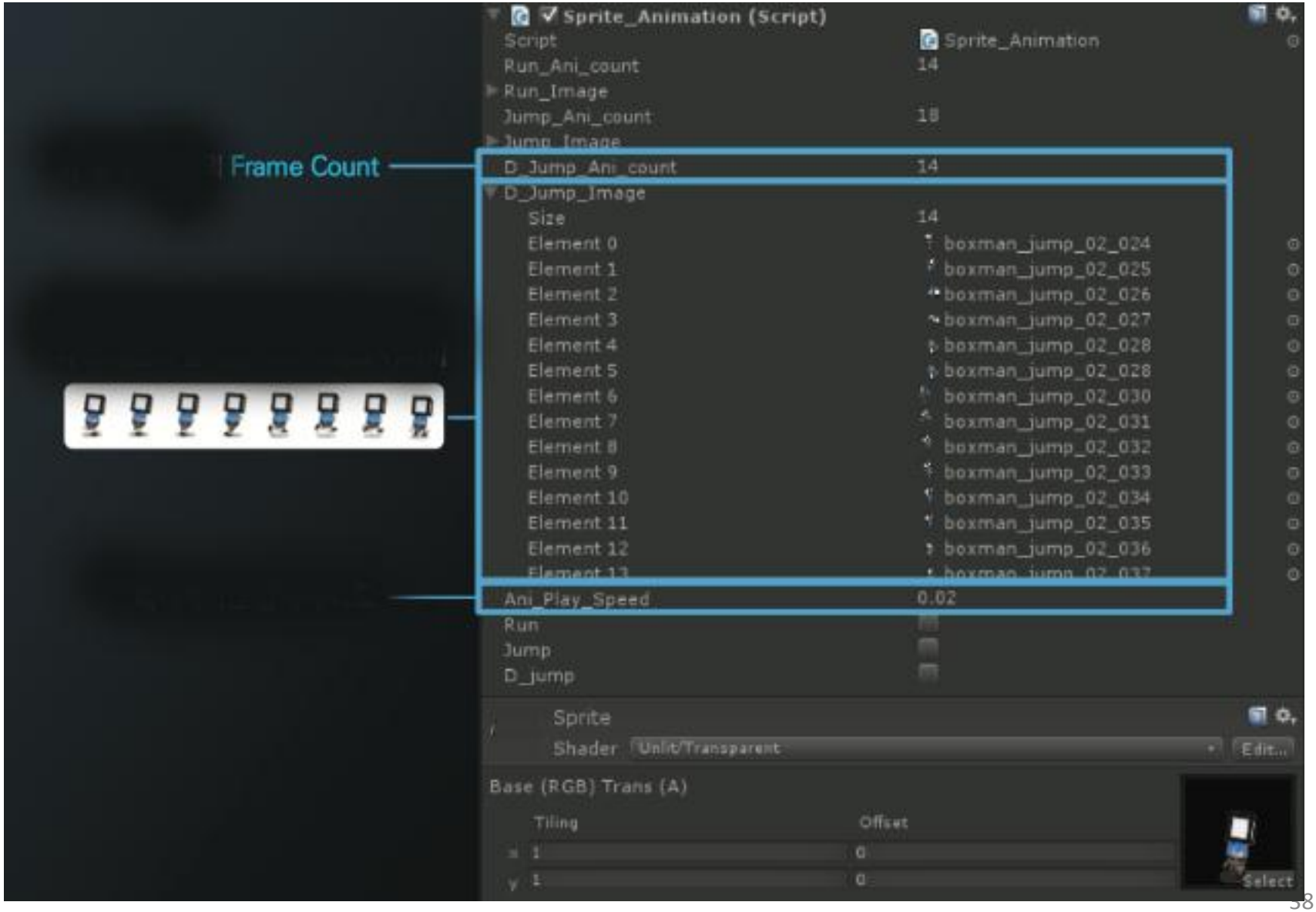
Kích hoạt Trigger để ăn coin



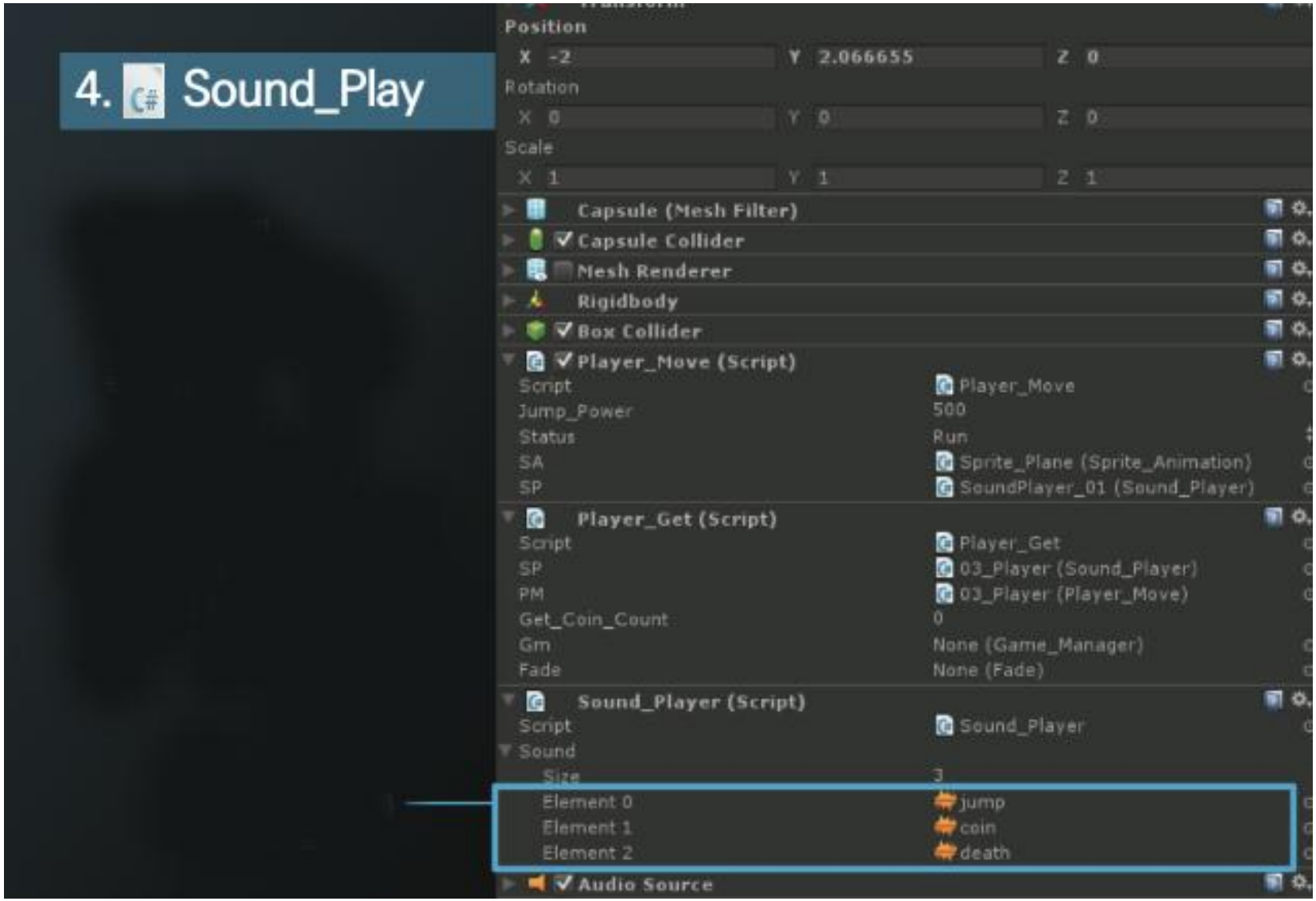
Animation cho Character



Thiết lập trạng thái của Animation



Cài đặt âm thanh theo trạng thái



Cài đặt Player_Move

- Định nghĩa các trạng thái cho PlayMove

```
public enum PlayerMoveStatus
{
    Run,
    Jump,
    DoubleJump,
    Die
};
```

- Khai báo các biến cần sử dụng:

```
public float Jump_Power;
public PlayerMoveStatus status;
public Sprite_Animation _SA;
public Sound_Player _SP;
```

Cài đặt Player_Move

- Định nghĩa hàm Run

```
void RUN ()
{
    //gán trạng thái cho character là Run
    status = PlayerMoveStatus.Run;
    if (_SA != null)
        _SA.Run_Play ();
}
```

- Định nghĩa hàm Jump

```
void JUMP ()
{
    //gán trạng thái cho character là Jump
    status = PlayerMoveStatus.Jump;
    rigidbody.AddForce (0, Jump_Power * 1.5f, 0);


    if (_SA != null)
        _SA.Jump_Play ();


    if (_SP != null)
        _SP.SoundPlay (0);

}
```

Cài đặt Player_Move

- Định nghĩa hàm nhảy kép (DoubleJump)

```
void DOUBLEJUMP ()
{
    //gán trạng thái cho character là nhảy kép
    status = PlayerMoveStatus.DoubleJump;
    rigidbody.AddForce (0, Jump_Power, 0);


    if (_SA != null)
        _SA.D_Jump_Play ();


    if (_SP != null)
        _SP.SoundPlay (0);
}
```

Cài đặt Player_Move

- Điều khiển Run, Jump bằng bàn phím

```
void KEYBOARD ()
{



    if (Input.GetButtonDown ("Jump")) {

        if (status == PlayerMoveStatus.Jump) {
            DOUBLEJUMP ();
        }


        if (status == PlayerMoveStatus.Run) {
            JUMP ();
        }
    }
}
```

Cài đặt Player_Move

- Điều khiển "chạm" qua màn hình mobile

```
void TOUCH ()
{

    if (Input.touchCount > 0) {
        if (Input.GetTouch (0).phase == TouchPhase.Began) {

            if (status == PlayerMoveStatus.Jump) {
                DOUBLEJUMP ();
            }

            if (status == PlayerMoveStatus.Run) {
                JUMP ();
            }
        }


    }
}
```

Cài đặt Player_Move

- Xử lý Collision

```
void OnCollisionEnter (Collision Get)
{
    if (status != PlayerMoveStatus.Die)
        RUN ();
}
```

- Hàm Update:

```
void Update ()
{
    KEYBOARD ();
    TOUCH ();
    rigidbody.WakeUp ();
}
```

Tổng kết

- Ngôn ngữ kịch bản trong Unity
- Tạo Script trong Assets
- Cấu trúc mặc định của Script
- Attach Scripts vào Objects trong môi trường
- Kết nối với đối tượng trong môi trường, sử dụng ‘tag’
- Sử dụng trình Debug Utility
- Thiết lập và kết nối Script

Fpt University

FPT POLYTECHNIC

THANK YOU!

www.poly.edu.vn