Làm Game 2D bằng Unity - Phần 5 - Thành phần vật lý và xử lý va chạm

20 February 2014 at 10:03

Chào các bạn,

Dạo này mình khá nhiều việc, nên việc viết bài hướng dẫn hơi bị gián đoạn, mong các bạn thông cảm (Hì :D).
Ở phần trước chúng ta đã tìm hiểu được các tạo Prefab và một số thao tác cơ bản với Prefab (xoá, thêm) trong quá trình game thực thi, và các thao tác cơ bản khác...

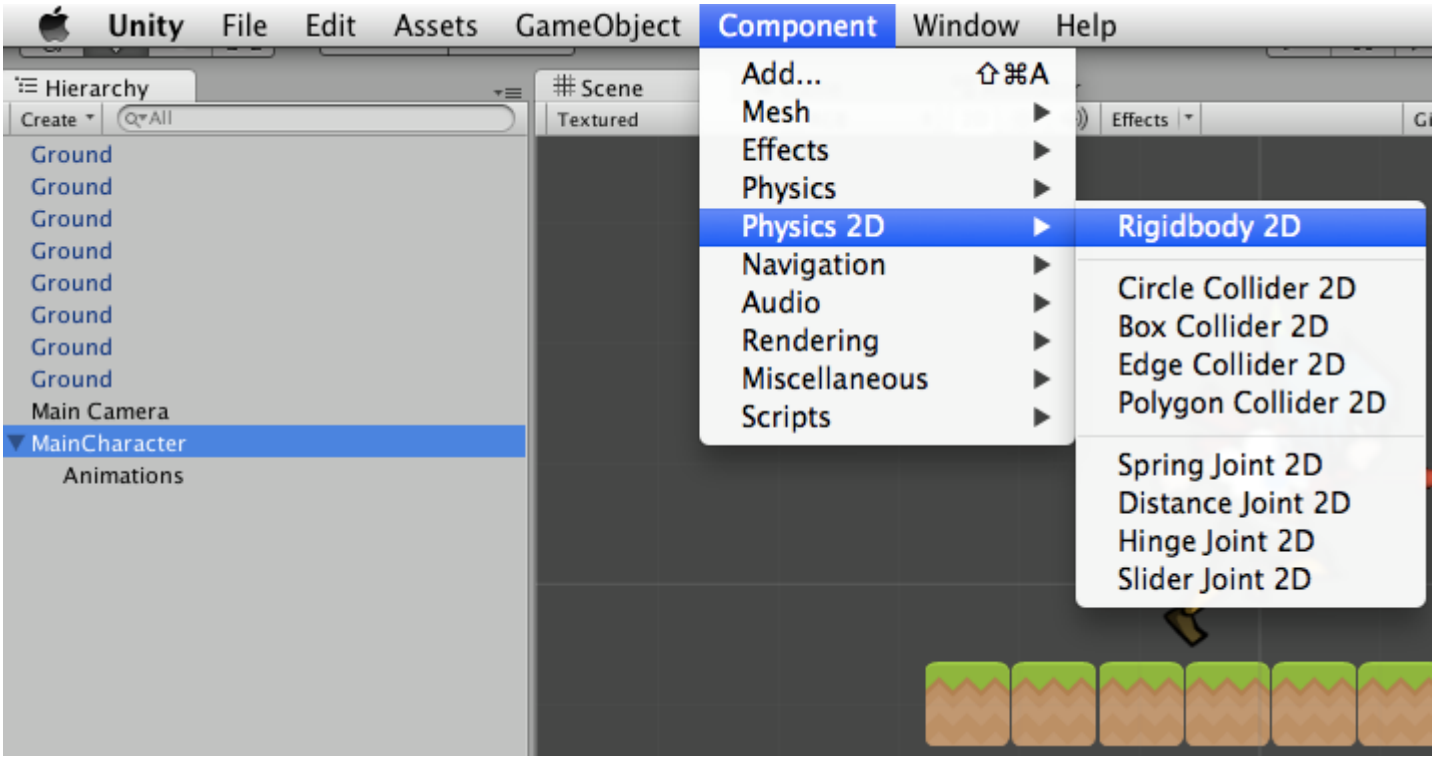
Tiếp theo hôm nay mình sẽ hướng dẫn cách thêm phần xử lý vật lý và va chạm vào đối tượng game.

Trước khi đi tiếp, nếu bạn nào chưa đọc các phần trước, ta có thể tham khảo lại ở đây
Link phần 4: https://www.facebook.com/notes/hội-lập-trình-viên-game-đà-nẵng/làm-game-2d-bằng-unity-phần-4-prefab-script-và-một-số-xử-lý-cơ-bản/242866329218970

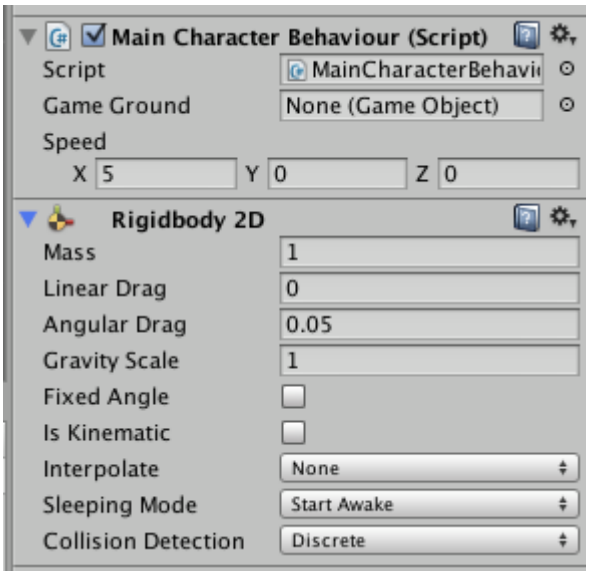
Link dự án mẫu phần 5: https://www.dropbox.com/sh/xu5rl7s3ssch2bk/e_Ny9i4LwW

III. Tạo các đối tượng cơ bản
1. Game Object
2. Sprite
3. Animation
4. Prefab
5. Script và điều khiển máy trạng thái
6. Thành phần vật lý và xử lý va chạm
a. Thêm thành phần vật lý (Physics 2D)

Thành phần vật lý hỗ trợ các thao tác về vật lý như: tác dụng lực, trọng lực trái đất, ma sát...
Để thêm thành phần vật lý ta làm như sau: ở Hierarchy, chọn đối tượng MainCharacter (đối tượng cần thêm) **/ Menu /Component / Physics 2D / Rigidbody 2D**



Ta sẽ thấy ở cửa sổ Inspector khi chọn đối tượng MainCharacter sẽ thêm một Component nữa là Rigidbody 2D như sau:



Một số giá trị cần lưu ý sau:

Mass: là khối lượng của đối tượng
Linear Drag: Hệ số ma sát của vật đối với chuyển động kéo.

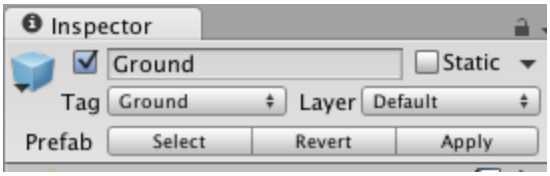
Angular Drag: Hệ số ma sát của vật đối với chuyển động quay
Gravity scale: giống như hệ số G trong vật lý (~ 9.81), chỉ sự ảnh hưởng của lực hút trái đất. Ta có thể đặt = 0, tức là không ảnh hưởng bởi lực hút trái đất.
Is Kinematic: loại bỏ tác dụng vật lý ra khỏi đối tượng, thường sử dụng với các đối tượng như tường, nền ...
Fixed Angle: Đối tượng luôn nằm một góc cố định. Không thay đổi khi tương tác vật lý.
rigidbody2D.AddForce(Vector2 f): phương thức tác dụng một lực vào đối tượng

Hoặc các bạn có thể tham khảo chi tiết, các thuộc tính và phương thức ở đây https://docs.unity3d.com/Documentation/ScriptReference/Rigidbody2D.html

Sau khi thêm thành phần vật lý vào, nhấn nút play để xem demo, ta sẽ thấy đối tượng sẽ từ từ rơi xuống (do Gravity scale > 0).

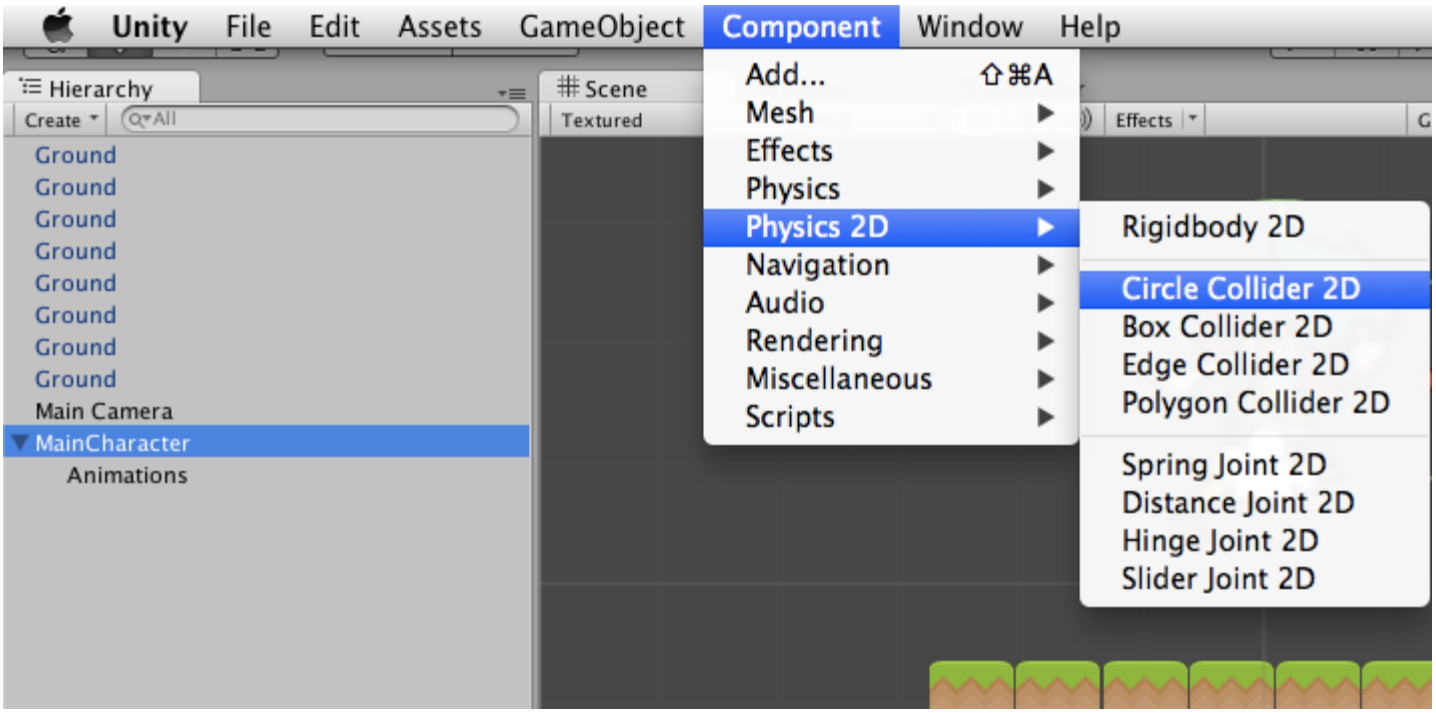
Tương tự, ta sẽ thêm thành phần vật lý cho đối tượng Ground, và đặt thuộc tính cho nó là**Is Kinematic** để làm nền.

Chú ý: Để có tác dụng cho tất cả các Prefab, sau khi thêm hoặc thay đổi bất cứ thuộc tính nào, ta nhấn Apply.

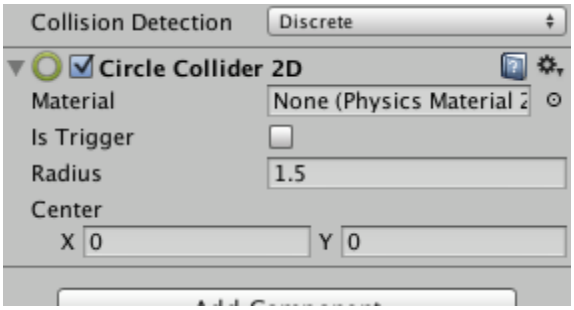


b. Thành phần xử lý va chạm

Ở Hierarchy, chọn đối tượng MainCharacter (đối tượng cần thêm) **/ Menu /Component / Physics 2D / Circle Collider 2D**



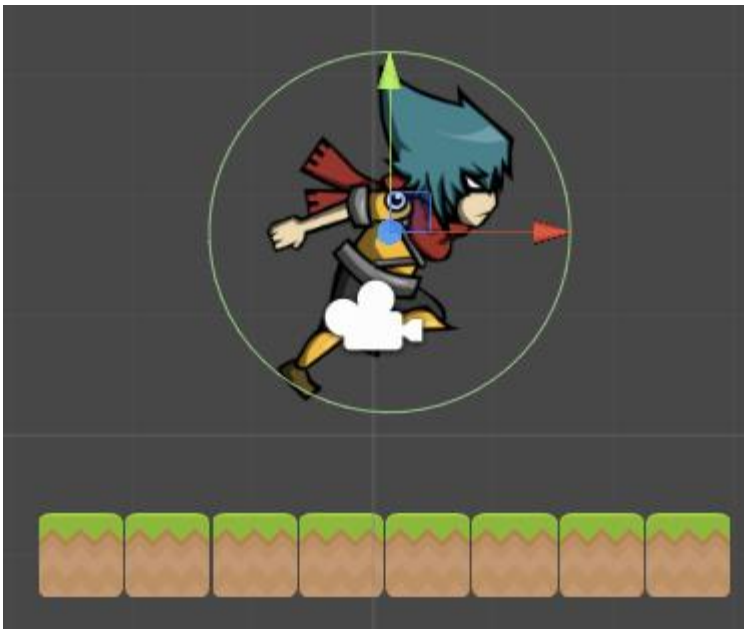
Đối tượng sẽ có thêm thuộc tính Collider:



Ta sẽ chọn tâm và bán kính để xác định vùng xử lý va chạm.

Thuộc tính: Is Trigger: nếu chọn thì đối tượng chỉ dùng để xác định va chạm mà không ảnh hưởng bởi tác động vật lý.

Kết quả ta sẽ thấy như sau:



Tiếp theo ta tiến hành thêm thành phần Collider cho đối tượng Ground, lần này ta sẽ chọn Box Collider 2D thay vì Circle Collier 2D. Các thuộc tính cũng tương tự, ta có thể chỉnh sửa hình chữ nhật để xác định vùng va chạm.

Chú ý: ta chọn Circle Collider 2D cho đối tượng MainCharacter để tránh trường hợp nền (cái đối tượng Ground, có độ cao không đều) nhấp nhô dẫn đến các không di chuyển được nhân vật MC hay còn gọi là bị Stuck.

Bây giờ ta nhấn nút play để test thì sẽ thấy đối tượng rơi xuống, gặp cái dãy đối tượng Ground thì đứng lại, và đối tượng bị nghiêng (Ta tưởng tượng có một chiếc bánh xe hình tròn rơi xuống mặt đất, nó sẽ lăn :D). Để tránh đối tượng nghiêng này chúng ta sẽ tick vào thuộc tính Fixed Angle của nhân vật MainCharacter.

Ta sẽ tìm hiểu thêm 3 phương thức:

```
void FixedUpdate()
{
    Debug.Log ("Fixed update");
}

void OnCollisionEnter2D(Collision2D other)
{
    //other.gameObject
    Debug.Log ("OnCollisionEnter2D with object has tag = " + other.gameObject.tag);
}
void OnTriggerEnter2D(Collider2D other)
{
    //other.gameObject
    Debug.Log ("OnTriggerEnter2D with object has tag = " + other.gameObject.tag);
}
```

```
void FixedUpdate()
{

}
```

--> Các tính toán, tương tác vật lý, chúng ta sẽ đặt trong hàm này, ví dụ như AddForce, etc (Chi tiết http://docs.unity3d.com/Documentation/ScriptReference/MonoBehaviour.FixedUpdate.html)

```
void OnCollisionEnter2D(Collision2D other)
{

}
```

--> Hàm này được gọi khi có hai đối tượng va chạm nhau.

```
void OnTriggerEnter2D(Collider2D other)
{

}
```

--> Hàm này được gọi khi có hai đối tượng va chạm nhau, trong đó có 1 hoặc cả hai đối tượng là Trigger.

Các bạn có thể tham khảo thêm chi tiết nhiều hàm khác ở đây: http://docs.unity3d.com/Documentation/ScriptReference/MonoBehaviour.html

Sau khi đã thêm phần tính toán va chạm và vật lý, ta sẽ cập nhập lại các điều khiển đối tượng bằng cách sử dụng tác dụng vật lý để làm đối tượng di chuyển, và nhảy.

Khai báo thêm hai thuộc tính movingForce và jumpForce trong MainCharacterBehavior để lưu trữ giá trị lực khi nhảy và khi di chuyển:

```
public Vector2 movingForce;
public Vector2 jumpForce;

protected Animator animator;
// Use this for initialization
void Start () {
    speed = new Vector3 (3, 0, 0);
    movingForce = new Vector2(10, 0);
    jumpForce = new Vector3(0, 50);
    animator = gameObject.GetComponent<Animator>();
}
```

Ở hàm Fixed Update:

```
void FixedUpdate()
{
    Vector3 localScale = transform.localScale;
    //Translation$
    if (Input.GetKey (KeyCode.LeftArrow))
    {
        rigidbody2D.AddForce(-movingForce);
        if (localScale.x > 0)
        {
            localScale.x *= -1.0f;
        }
    }

    if (Input.GetKey(KeyCode.RightArrow))
    {
        rigidbody2D.AddForce(movingForce);
        if (localScale.x < 0)
        {
            localScale.x *= -1.0f;
        }
    }

    transform.localScale = localScale;
}
```

Bây giờ các bạn có thể xử lý cho đối tượng nhảy lên khi nhấn nút up/down.
Vậy là chúng ta đã biết cách thêm và sử dụng thành phần vật lý và xử lý va chạm, phần tiếp theo ta sẽ tìm hiểu tiếp cách hiển thị text, thông tin và cách chuyển màn chơi. Hay chuyển Scenes.

Link của phần tiếp theo: https://www.facebook.com/notes/hội-lập-trình-viên-game-đà-

nẵng/làm-game-2d-bằng-unity-phần-6-text-particle-system-và-chuyển-đổi-màn-chơi/247194832119453