Hướng dẫn tạo Raycast trong Unity 3D

Chào các bạn hôm nay mình sẽ hướng dẫn các bạn sử dụng Raycast trong unity , đây là một kỹ thuật khá cơ bản vì vậy bạn nào mới làm unity cũng nên biết . Ứng dụng của nó thì có nhiều mình ví dụ như để bắt va chạm của viên đạn với vật thể nào trong game hoặc đơn giản là để biết tay của bạn chạm vào vật nào trong game ...

Vậy Raycast là gì ?

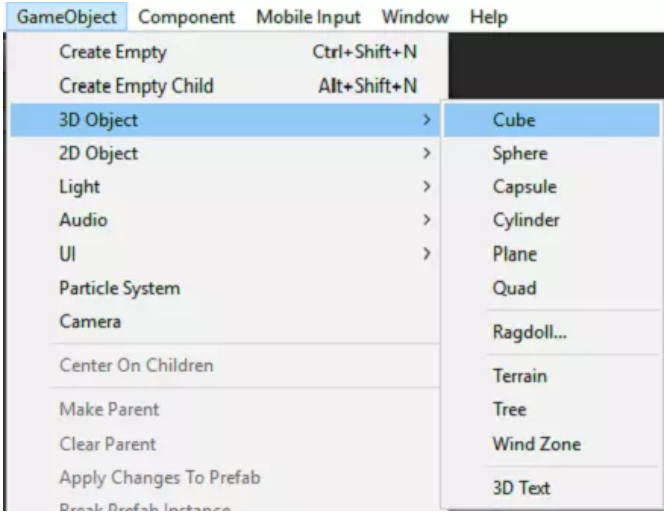
Raycasting thường được sử dụng trong những việc như xác định đường ngắm của người chơi hoặc AI, nơi một viên đạn sẽ đi qua, tạo ra tia laser và nhiều hơn nữa. Một raycast về cơ bản là một tia mà được gửi đi từ một vị trí trong không gian 3D hoặc 2D và di chuyển theo một hướng cụ thể. Unity đã xây dựng chức năng trong đó bạn có thể sử dụng để thực hiện Raycast trong trò chơi của bạn.

Hướng dẫn dùng Raycast

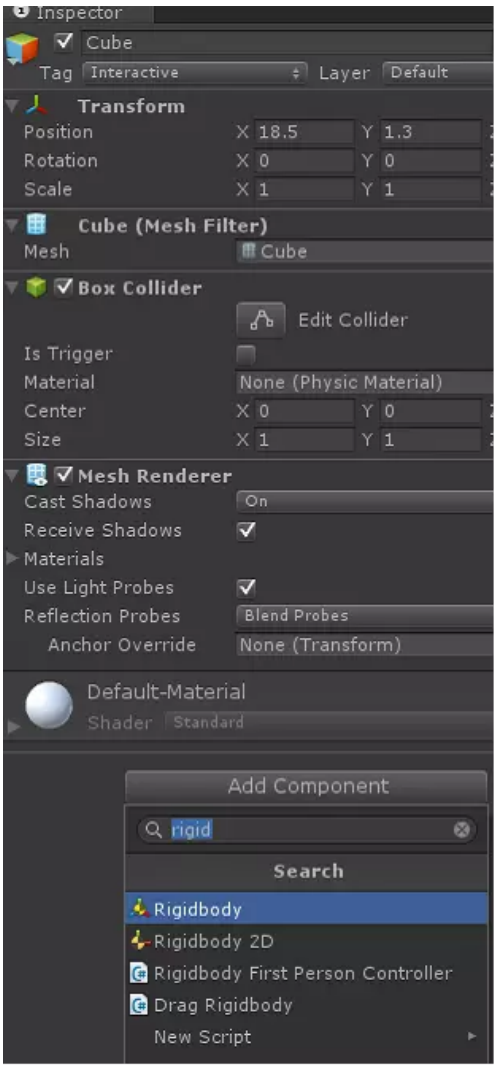
Trong bài này, tôi sẽ đưa ra ví dụ về làm thế nào để thực hiện một Raycast trong Unity 3D và cách sử dụng các thông tin của Raycast để xác định xem một đối tượng có nằm trong đường ngắm của người chơi không và nếu có chúng ta có thể tương tác với đối tượng đó.

Bước 1: Trước hết, tôi sẽ giả định rằng bạn đã có một scene với một bộ điều khiển nhân vật mà bạn có thể sử dụng. Nếu bạn không có thì hãy tạo ra một cảnh mới, tạo ra một plane mà bạn có thể đi bộ trên đó, và đặt một bộ điều khiển nhân vật trong scene. Nếu bạn không có một bộ điều khiển, hãy import bộ asset tiêu chuẩn assets / characters package và kéo và thả một trong các bộ điều khiển prefab vào scene của bạn.

Bước 2: Một khi bạn đã thiết lập scene cơ bản của bạn, chúng ta sẽ tạo ra một khối lập phương bằng cách vào GameObject → 3D Object → Cube, như dưới đây:



Di chuyển khối hộp tới nơi nào đó gần với bộ điều khiển của bạn để chúng ta có thể nhanh chóng và dễ dàng kiểm tra bất kỳ chức năng chúng ta đang làm việc hay không. Một khi các khối hộp được đặt ra, chúng ta sẽ gắn một tag tùy chỉnh cho nó. Chúng ta sẽ sử dụng tag để xác định những đối tượng chúng ta đang tìm kiếm khi Raycast của chúng ta truy cập nó. Hãy nhớ rằng bạn có thể làm điều này với một tên của đối tượng hoặc bất kỳ biến khác nhau nào.



Bước 3:

Để đơn giản, chúng ta sẽ sử dụng một tag để xác định những gì chúng ta đang tìm kiếm. Để tạo ra một tag tùy chỉnh, chọn khối lập phương trong scene của bạn, và trong inspector, nhấp vàoTag và chọn "Add Tag ..."

Trong quản lý thẻ, nhấp vào biểu tượng + để tạo ra một tag mới. Hãy gọi tag này là "Interactive".

Bước 4: Khi tag được tạo ra, nhấp chuột vào Cube lần nữa và chọn tag là Interactive

Bước 5: Tiếp theo, nhấn vào khối hộp và thêm thành phần rigidbody bằng cách sử dụng cửa sổ Add Component

Bước 6: Bây giờ chúng ta thực hiện các raycast. Chọn main camera của bạn trong scene của bạn và thêm một thành phần C # Script mới cho nó gọi là LineOfSight:

Nội dung trong file script như sau :

```
using UnityEngine;
using System.Collections;

public class LineOfSight : MonoBehaviour {

    private RaycastHit vision; //Used for detecting Raycast collision
    public float rayLength; //used for assigning a length to the raycast
    private bool isGrabbed; //used so we know whether or not we're holding an object
    private Rigidbody grabbedObject; //used to assign the object we're looking at to a variable we can use

    void Start ()
    {
        rayLength = 4.0f;
        isGrabbed = false;
    }

    void Update ()
    {
        //This will constantly draw the ray in our Scene View so we can see where the ray is going
        Debug.DrawRay(Camera.main.transform.position, Camera.main.transform.forward * rayLength, Color.red, 0.5f);

        //This statement is called when the Raycast is hitting a collider in the scene
        if(Physics.Raycast(Camera.main.transform.position, Camera.main.transform.forward, out vision, rayLength))
        {

            //determine if the object our raycast is hitting has the "Interactive" tag
            if(vision.collider.tag == "Interactive")
            {
                Debug.Log (vision.collider.name); //Output the name of the object our raycast is hitting to our console

                //If our user is looking at an interactive object, allow them to press E and pick it up but only if we aren't holding something
                if(Input.GetKeyDown (KeyCode.E) && !isGrabbed)
                {
                    grabbedObject = vision.rigidbody; //assign grabbedObject to the object we're looking at
                    grabbedObject.isKinematic = true; //set the rigidbody to kinematic so it ignores physics
                    grabbedObject.transform.SetParent (gameObject.transform); //set grabbedObject's parent to our camera
                    isGrabbed = true; //we are now grabbing an object so set this to true
                }
                //if we have an object already and user presses E, drop the object
                else if(isGrabbed && Input.GetKeyDown(KeyCode.E))
                {
                    grabbedObject.transform.parent = null; //set grabbedObject's parent back to nothing
                    grabbedObject.isKinematic = false; //allow grabbedObject to interact with physics again
                    isGrabbed = false; //we are no longer grabbing anything.
                }
            }
        }
    }
}
```

Khởi tạo biến trong Script: Điều đầu tiên trong script của chúng ta đang làm là khai báo 4 biến : RaycastHit, float, bool, vật rắn. Chúng ta sẽ sử dụng tất cả các biến này để quản lý logic. Trong hàm Start, chúng ta đăng ký một biến boolean và một biến float với giá trị mặc định. Điều này sẽ xảy ra khi game bắt đầu.

Đây là dòng đầu tiên trong hàmUpdate :

Debug.DrawRay(Camera.main.transform.position, Camera.main.transform.forward * rayLength, Color.red, 0.5f);

Tất cả điều này là để vẽ một đường màu đỏ trong Scene khi game đang chạy. Tham số đầu tiên xác định vị trí mà nó bắt đầu, tham số thứ hai là hướng di chuyển, tham số thứ ba xác định màu , và tham số thứ tư quy định cụ thể thời gian mà các đường thẳng được vẽ trên màn hình. Chúng ta làm điều này chủ yếu cho mục đích debug. Nó cho phép chúng ta thực sự thấy raycast của chúng ta trong Scene nhìn như thế nào. Chúng ta có thể nhìn thấy nó trong Game view bằng cách nhấn vào "Gizmo" tùy chọn trong cửa sổ game view như dưới ở đây:

Tiếp theo, chúng ta hãy nhìn vào những gì thực sự gây ra raycast :

if(Physics.Raycast(Camera.main.transform.position, Camera.main.transform.forward, out vision, rayLength))

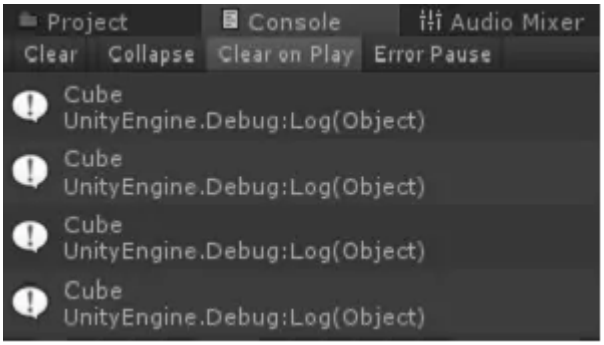
nếu Raycast chạm một cái gì đó. Tham số đầu tiên là vị trí raycast bắt đầu , ở đây chúng tôi xác định là vị trí camera . Tham số thứ hai là hướng của raycast ở đây chúng tôi xác định là hướng về phía trước của camera. Điều này đảm bảo rằng bất cứ nơi nào chúng ta nhìn, các Raycast sẽ là tốt. Tham số thứ ba là tham số thông tin hit(điểm va chạm). Nó sẽ cung cấp thêm thông tin về các vụ va chạm của raycast mà chúng tôi chỉ định. Trong ví dụ này, chúng tôi chỉ định biến RaycastHit gọi là tầm nhìn như là đối số thông tin hit. Điều này có nghĩa rằng biến raycasthit của chúng ta sẽ có những thông tin về vụ va chạm với đối tượng . Cuối cùng, tham số thứ tư là chiều dài của raycast, ở đây chúng ta đặt là rayLength. Chúng ta gán giá trị cho rayLength trong hàm Start. Đó là một biến publish nên chúng tôi có thể sửa đổi nó trong inspector, nếu chúng ta muốn kiểm tra độ dài .

if(vision.collider.tag == “Interactive”)

Nếu đối tượng được gắn tag là "Interactive", mà chúng ta thiết lập trước đó, sau đó chúng tôi sẽ thực hiện một cái gì đó.

Debug.Log (vision.collider.name);

Dòng này chỉ đơn giản là xuất ra tên của các đối tượng, chúng ta có thể nhìn thấy nó trong cửa sổ console. Bạn sẽ thấy khi bạn là đủ gần và nhìn vào khối lập phương trong scene của bạn tên của khối lập phương sẽ được hiển thị trên giao diện điều khiển như vậy:



Tiếp theo, chúng ta sẽ có thêm một lệnh if lồng vào nữa nến chúng ta tìm kiếm được một đối tượng tương tác :

if(Input.GetKeyDown (KeyCode.E) && !isGrabbed)

Nếu player của chúng tôi đang tìm kiếm một đối tượng tương tác, điều kiện này này sẽ cho phép người chơi nhấn phím E trên bàn phím và thực hiện một cái gì đó nhưng chỉ khi chúng ta đang KHÔNG "grabbing" một cái gì đó. Vì vậy, một khi player của chúng ta nhấn E và được phép nhận một đối tượng, chúng ta thực thi mã này:

grabbedObject = vision.rigidbody;

grabbedObject.isKinematic = true;

grabbedObject.transform.SetParent (gameObject.transform);

isGrabbed = true;

Chúng ta gắn cho grabbedObject một rigidbody do đó raycast của chúng ta có thể tương tác. Hãy ghi nhớ điều này chỉ xảy ra nếu chúng ta nhìn vào một cái gì đó gắn tag là "Interactive". Tiếp theo, chúng ta đặt grabbedObject là một đối tượng kinematic điều này sẽ vô hiệu hóa các tương tác vật lý trên đối tượng. Điều này sẽ cho phép người chơi của chúng ta có thể chọn và di chuyển đối tượng mà không có nó bị ảnh hưởng bởi phép quay, trọng lực, vv Sau đó, chúng ta thiết lập các parent của grabbedObject là main camera. Điều này cho phép các đối tượng nắm lấy để di chuyển tương đối so với main camera . Cuối cùng, chúng ta thiết lập isGrabbed là true để trò chơi của chúng ta biết rằng chúng ta đang sở hữu một đối tượng.

else if(isGrabbed && Input.GetKeyDown(KeyCode.E))

nếu người chơi đang nắm giữ một đối tượng và nếu họ nhấn phím E. Nếu cả hai đều đúng, sau đó chúng ta thả các đối tượng bằng cách thực hiện các mã sau :

grabbedObject.transform.parent = null;

grabbedObject.isKinematic = false;

isGrabbed = false;

Đầu tiên, chúng ta đặt cha của grabbedObject trở lại để nó không là con của camera . Sau đó, chúng ta set nó là non-kinematic để nó tương tác với vật lý . Cuối cùng, chúng ta thiết lập isGrabbed là false để chúng ta có thể nhận một đối tượng khác, nếu chúng ta muốn.

Bước cuối cùng: Tại thời điểm này, bạn sẽ có thể chạy trò chơi và đi bộ đến khối lập phương của bạn và nhìn vào nó. Bạn sẽ thấy hiệ ra tên của nó trên giao diện điều khiển của bạn, và bạn sẽ có thể nhấn E để nhặt nó lên và di chuyển xung quanh với nó. Sau đó bạn sẽ có thể nhấn E lần nữa để thả nó ra .