by yselim

UNITY GÜNLÜĞÜ

LERP VE RAYCAST KULLANIMI

LERP Kullanımı

Lerp fonksiyonunun çalışması çok basit bir işleyişe sahip:

a ve b sayılarının arasındaki bir değeri döndürmek. Bu değerin a veya b'den ne kadar uzak olacağını ise üçüncü parametre belirliyorMathf.Lerp(0, 100, 0.5): 0 ile 100 sayılarının tam ortasındaki sayıyı (50) döndürür.

Mathf.Lerp(0, 100, 0.75): Döndürülen sayının 0'a uzaklığı %75 iken 100'e uzaklığı %25 olacak, yani 75 döndürülecek.

Mathf.Lerp(100, 500, 0.1): Döndürülen sayının 100'e uzaklığı %10 iken 500'e uzaklığı %90 olacak. Aradaki uzaklık 400 olduğuna göre bu uzaklığın %10'u 40'tır. O halde döndürülen sayı 100+40=140 olacak.

NOT: Üçüncü parametre 0.0 ile 1.0 arasında değilse Unity tarafından otomatik olarak ya 0 ya da 1 yapılır.

NOT1: *Mathf.Clamp fonksiyonu: verilen aralıkta kalınmasını sağlıyor! Orn:-3 +3 aralığında kal*

LERP Kullanımı

Update ilk kez çağrıldığında a'nın değeri 0 ile 100'ün ortasındaki sayı, yani 50 olacak. Update ikince kez çağrıldığında a'nın değeri 50 ile 100'ün arasındaki sayı, yani 75 olacak. Sonraki frame'de 87.5 diyerekten böyle böyle a sayısı b sayısına yaklaşacak. Fonksiyondaki 0.5'i artırırsak a daha hızlı yaklaşır, azaltırsak daha yavaş yaklaşır.

```
public float a= 0;
public float b= 100;


Update()
{
    a = Mathf.Lerp( a, b, 0.5 );
}
```

LERP Kullanımı (Örnek)

UI button olsun ve bu buton ekranda fare imlecini takip etsin. Burada **a** değerimiz butonun pozisyonu, **b** değerimiz ise fare imlecinin pozisyonu olduğu taktirde buton fareyi takip eder. O halde sahnenizde **GameObject-UI-Button** yoluyla bir buton oluşturup butona şu kodu verin:

```
public class butonTakip : MonoBehaviour {

    public Vector3 butonKonum;
    public Vector3 imlecKonum;
     float fark;
// Use this for initialization
void Start () {

}

void Update () {
        butonKonum = transform.position;
        imlecKonum = Input.mousePosition;

        if (imlecKonum.x - butonKonum.x < 1)
            butonKonum = imlecKonum;
        transform.position=Vector3.Lerp(butonKonum, imlecKonum, Time.deltaTime);
} }
```

Mathf.Lerp

Vector3.Lerp

FPS Light Kullanımı

```
public class isikKontrol : MonoBehaviour {
   public Light isik;
// Use this for initialization
void Start () {

        isik = GetComponent<Light>();
        isik.enabled = false;
}

// Update is called once per frame
void Update () {
        if (Input.GetMouseButtonDown(0))
            isik.enabled = !isik.enabled;

    }
}
```

FPS Light Kullanımı

```
public class isikKontrol : MonoBehaviour {

    public Light isik;
    // Use this for initialization
    void Start () {
        isik = GetComponent<Light>();
        isik.intensity = 100;
        isik.enabled = false;
    }
    void Update ()
     {
         if(Input.GetMouseButton(0))
        {
            Debug.Log("mouse tıklandı");
            isik.enabled = false;
            //isik.enabled = !isik.enabled;
           // StartCoroutine(bekle());
        }

        if(Input.GetMouseButtonDown(1))
        {
            isik.enabled = true;
        }

    }
```

```
IEnumerator bekle()
    {
        if (isik.intensity < 90)
        {
            yield return new WaitForSeconds(.1f);
            isik.intensity += 10;

        }
    }
```

RAYCAST Kullanımı

Oyunun içinde istediğimiz doğrultuda ve uzunlukta görünmez düz bir çizgi cizdiririz ve bu çizgiyle temasta bulunan birşey olursa istediğimiz olayları yaptırabiliriz.

The purpose of the ray (vector) is to determine if it intersects with any colliders or other game objects. This function is most commonly used in first-person shooter games, where the main camera (the character) is the ray's origin (starting point) and the ray is cast to interact with other objects in the game that may come in contact with the ray.

RAYCAST Kullanımı

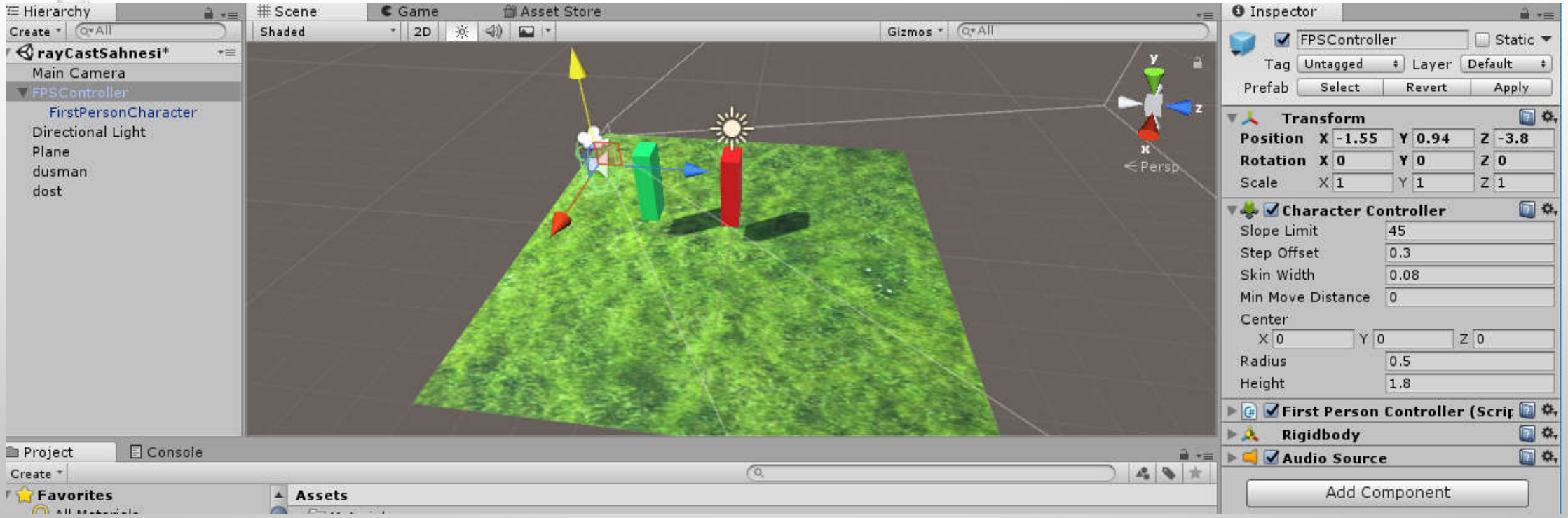
Physics.Raycast(baslangicKonumu : Vector3, yon : Vector3, raycastHitDegiskeni (İsteğe bağlı): RaycastHit, raycastUzunlugu: float)

//Raycast(baş,yön,hit,kaç mt)

Başlangıç konumundan ilgili yöne ilgili uzunlukta bir raycast ışını yollar ve bu raycast ışını bir objenin collider'ine temas ederse fonksiyon "true" değerini döndürür.Eğer raycast ışını bir şeye çarpmazsa fonksiyonun döndürdüğü değer "false" olur.Ayrıca isteğe bağlı olarak bir değişken oluşturup tipini "RaycastHit" yaparsan ve o değişkenin adını metoddaki ilgili kısma yazarsan raycast ışınının temas ettiği obje hakkında çok detaylı bilgiler alabilirsin.

Örneğin objeyle raycast'in başlangıç noktası arasındaki uzaklık, objenin adı vb. Ayrıca objenin "gameObject" i sayesinde onun üzerinde istediğin değişikliği yapabilirsin.

RAYCAST Kullanımı (Uygulama)



Sahne: Plane , Dost ve Düşman isimli iki Cube. Birde FPC(first person character) eklenmiş. FPC Asset/Import Package/Characters paketinden.

Raycast Düşman isimli objeye çarpınca Düşmanı yok edecek. Script FPS Character/Controller'a atanır

by yselim

RAYCAST Kullanımı (Uygulama)

```
public class raycastKontrol : MonoBehaviour
{
    RaycastHit carpmaBilgisi;
    public Text saglikTxt;
    public static int saglikdeger = 100;
    // Start is called before the first frame update
    void Start()
    {

    }
    private void Update()
    {
        Debug.DrawRay(transform.position, transform.TransformDirection(new Vector3(0, 0, 1) * 10), Color.red);
    }

    private void FixedUpdate()
    {
            if(Physics.Raycast(transform.position,transform.TransformDirection(new Vector3(0,0,1)),out carpmaBilgisi,10)
&& carpmaBilgisi.collider.gameObject.name=="dusman" && Input.GetMouseButtonDown(0))
        {
            saglikdeger -= 10;
            saglikTxt.text = saglikdeger.ToString();
            if (saglikdeger <= 0)
            {
                Destroy(carpmaBilgisi.collider.gameObject);
            }

        }
    }
}
```

▲ 12 of 16 ▼ bool Physics.Raycast(Vector3 **origin**, Vector3 direction, out RaycastHit hitInfo, float maxDistance)