**3. Hafta Raporu – Aleks Dulda 21360859025**

**Enemy prefab oluşturma:**

metin, ekran görüntüsü, yazılım, bilgisayar simgesi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Hierchy kısmından 3D obje olarak Cube öğesini ekliyoruz. Bunu Mouse sağ tıkını kullanıp -> 3D obje -> Cube olarak seçebilirsiniz.

Oluşturduğumuz Cube öğesini enemy olarak adlandırabilirsinz.

Oluşturduğumuz Cube’ü Hierchy den Project penceresinde önceden oluşturduğumuz "Prefabs” klasörüne sürüklenerek prefab haline getirdik

**metin, ekran görüntüsü, diyagram, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Bu şekilde prefabımızı oluşturmuş olduk.

**Enemy hareketini sağlayan kodun yazılması:**

**metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

private Player\_Sc player; //Kalıtım alındı ve Playerin dinamik olarak ekranın dışına çıkmaması için gereken verileri player\_sc kodlarından çekmiş olduk.

metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu kısımda

transform.Translate(Vector3.down \* speed \* Time.deltaTime); komutuyla enemy objemizin hareketi sağlandı.

İf koşulu sayesinde oluşacak enemy onjesinin rastgele belirli sınırlar içersinde random olarak oluşmasını sağladık.

Sınırları kalıtım aldığımız Player\_Sc scriptinde belirlediğimiz sınırları kullandık.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu**RigidBody ve Collision component’lerinin eklenmesi:**

Collider ve Rigidbody kullanarak objelerin hareketi ve gerekli temasların kontrollerinin sağlanması için bu componontleri ekledik.

Collider kısmında “In Trigger” check box’ını işaretliyoruz. Ki objeler birbirinin içinden geçebilsin ve temasları kontrol edilebilsin.

Rigidbody kısmında ise “Use Gravity” kapattık ve “Freeze Position” kısmından X ile Z kordinatlarını Kilitledik ki sadece Y ekseninde hareket edebilsin.

Bu Component’leri Inspector penceresinin en alt kısmında bulunan “Add Component” kısmından ekleyebilirsiniz.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

**OnTriggerEnter fonksiyonu ve çarpışmaların tespiti ile gerekli aksiyonların alınması:**

**metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Bu kısımda “OnTriggerEnter” fonksiyonu ile Objelere eklediğimiz “Tag” etiketleri sayesinde objelerin birbirine olan etkileşimlerini kontrol ederek gerekli koşullara göre işlemler sağlanmakta.

İlk koşulda Player ile Enemy temasında oluşacak durum mevcuttur. Player ile Enemy temasında Player’ın canlarından 1 birim eksilmesi için “Damage” fonksiyonu çağrılmıştır ve Player’i yok etmektedir

Diğer koşulda ise Lazer ile Enemy karşılaşmasında oluşacak durum mevcuttur. Lazer ile Enemy temasta bulununca **önce** lazeri (çünkü enemy önce yok olursa kodların devamı gelmez) sonra Enemy yok edilir.

metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu**Oyuncu hasar alma işlemlerinin yapılması:**

Oyuncunun hasar alabilmesi fonksiyonunu Player\_Sc dosyasında bu Damage fonksiyonu ile yapılmıştır.

Bu fonksiyon bir önceki sayfada anlatılan koşullar içerisinde çağırılıp player’in canını bir birim azaltmakta ve eğer player’in canı kalmaz ise oyunu bitirmektedir.

Bu şekilde bu hafta için istenenler yapılmış oldu.

**Kaynakça**

[**https://en.wikipedia.org/wiki/Unity\_(game\_engine)**](https://en.wikipedia.org/wiki/Unity_(game_engine))

[**https://chatgpt.com/**](https://chatgpt.com/)

[**https://docs.unity3d.com/Manual/index.html**](https://docs.unity3d.com/Manual/index.html)

**Github link**

[**https://github.com/AleksDulda/GamePrograming**](https://github.com/AleksDulda/GamePrograming)

**https://github.com/AleksDulda/GamePrograming/tree/main/Rapor/Week\_3**