**Oyun Geliştirme Ödevi**

Cem Topuz

19240001452

Oyunun;

İsmi: Slime Jump

Türü: Endless Runner

Teması: Yukarıya doğru zıplayıp tırmanarak postaları teslim etmeye çalışan postacı bir slime’ı oynuyoruz.

Oynanışı: Oyun başladığında karakterimiz yerde durmaktadır, karakter ekranın sağına ve soluna dokunularak hareket ettirilir. Karakter başlangıçtaki zeminde olduğu sürece ekranın ortasına dokunarak zıplatılabilir. Karakter ilk platformun üstüne geldiği andan itibaren kendi otomatik zıplamaya başlar ve artık amaç ekranın sağına ve soluna dokunarak karakterimizi platformların üstünde tutmaktır (platformlara altından çarpmaz sadece üstlerine çıktıktan sonra çarpışma gerçekleşir). Karakter ekran sınırından aşağı düşerse oyunu kaybeder. Kaybedildiğinde çıkan ekrandan tekrar oynaya basarak baştan başlayabilirsiniz. Ekranın sol üst kısmında yükseklik sayacı var ve kaybedildiği andaki yüksekliği kaybetme ekranında yazar.

Açıklama: Oyun içerisindeki bütün çizimleri ve animasyonları kendim yaptım, hiçbir hazır asset kullanmadım. Zamanım yetmediği için; oyuna ses ekleyemedim (önceden yaptığım başka projelerde ses kullanmıştım yani ses ekleme tecrübem var), ilk defa bir oyunu build alıyorum bütün ekranlara uyumlu hale getirmeyi şimdilik yapamadım o yüzden bazı görüntü ve oynanış hataları olabilir (örn: sağ ve sol ekran limitleri fazla ileride kaldığı için karakter ekrandan çıkabilir.), oyuna eklemek istediğim bütün mekanikleri ekleyemedim daha fazla mekanik ve oynanış elementi ekleme planım var. Farkettiğim eksikler bunlar.

**Hangi Kodlar Ne İşe Yarıyor?**

playerController.cs

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada playerın iki farklı objeyle çarpışma kontrolünü sağlıyorum;

“Ground” tagına sahip bir objeyle çarpıştığında zıplama animasyonunu aktif hale getiriyor ve playerın rigidbodysinin velocity değerini arttırarak zıplamasını sağlıyor.

Playerın yerde olduğunu anlamak için “Floor” tagına sahip objeyle temas halindeyken onFloor isimli boolean değişkenini true yapıyor ve animatör içerisinde player yerdeyken oynayacak animasyonların aktif olmasını sağlıyor.

playerController.cs

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

Açıklama otomatik olarak oluşturuldu

Burada ilk önce ekrana anlık dokunan parmak sayısını alıyorum ve eğer bu sayı 0’dan büyükse “touch” isminde bir Touch değişkeni oluşturuyorum ve o an girdisi alınan dokunuşa eşitliyorum ve class içerisinde oluşturduğum “touchPosition” isimli vector3 değerimi touch değişkeninin world pointe dönüştürülmüş pozisyonuna eşitliyorum.

Eğer karakter zemindeyse ve ekranda -0.5 ve 0.5 kordinatları arasında bir yere dokunulduysa karakterin zıplamasını, ve zıplama animasyonunun devreye girmesini sağlıyor.

playerController.cs

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Zıplama animasyonunu önceden belirlediğim süre boyunca aktif kalmasını sağlayan ve süre bitince animasyonu iptal eden kodlar.

playerController.cs

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Ekranın soluna dokunulduğunda playera sola doğru bir hız veriyor, ekranın sağına dokunulduğunda playera sağa doğru bir hız veriyor.

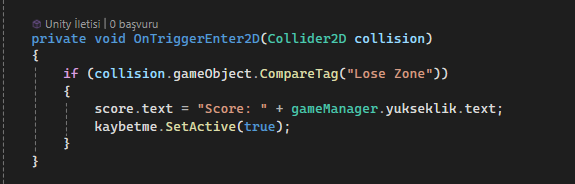
playerController.cs

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Eğer ekranın soluna dokunulduysa player objesinin sprite renderer bileşeninin flipX özelliğini aktif ederek karakterin sola dönmesini sağlıyor ve koşma animasyonunu başlatıyorum. Ve aynısını sağa gitme durumu için yapıyorum. Eğer ekrana hiç dokunulmuyor ise koşma animasyonunu iptal ediyorum.

playerController.cs



Player kaybetme alanına girdiği an ekranda yükseklik scoreunun yazmasını ve kaybetme isimli UI objesinin aktifliğini açıyorum bu şekilde kaybetme ekranı belirmiş oluyor.

gameManager.cs

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Kaybetme ekranındaki tekrar oyna tuşuna özellikleri vermek için gerekli fonksiyonu yazıyorum:

currentScene isminde oluşturduğum değişkene an aktif olan sahnenin indeksini eşitliyorum ve aynı sahneyi tekrar çağırarak oyunun tekrar başlamasını sağlıyorum.

gameManager.cs

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Start içerisinde gerekli objelerin tanımlamalarını yapıyorum ve playerın ilk yüksekliğini alıyorum.

Update içerisinde sürekli playerın yüksekliğinden ilk yüksekliği çıkartıyorum ki doğru bir yükseklik değeri elde edebileyim. Sonrasında yüksekliğin daha yavaş artması için yükseklik değerini ikiye bölüyorum.

Devamında player max yüksekliğinden daha yükseğe çıktığı anlarda sol üstte bulunan yükseklik bilgisini güncelliyorum. Bu şekilde score kısmını oluşturmuş oluyorum.

cameraController.cs

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Öncesinde playerın transformuna eşitlediğim target isimli transformun y değeri kameranın transformunun y değerinden büyükse yani player kameradan daha yüksekse kameranın playerı smooth bir şekilde takip etmesini sağlıyorum.

ground.cs

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

Açıklama otomatik olarak oluşturuldu

Burada önce oyun başladığında player objesini alıyor devamında ground objesi playerdan yukarıdaysa ground objesinin colliderının isTrigger özelliğini aktif ediyorum ki çarpışma gerçekleşmesin. Eğer ground objesi playerdan aşağıdaysa isTrigger özelliğini kapatıyorum.

ground.cs

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

Açıklama otomatik olarak oluşturuldu

Platformlar kaybetme alanına girdiği an belirlediğim iki aralık arasından random bir değer seçiliyor. Sonrasında ekrandan çıkan platformu en üste seçilen random x pozisyonunda ışınlıyorum. Bu şekilde sahnedeki 4 platform sürekli random x pozisyonunda ışınlanarak sonsuz bir oyun oluşuyor. Sürekli objeyi destroy edip instantiate etmek yerine objeleri ışınladığım için oyun daha optimize çalışıyor.

sceneManager.cs

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Start sahnesindeki “OYNA” ve “ÇIKIŞ” butonlarında kullanmak için fonksiyonlar oluşturuyorum. İlkine tıklandığında seçili sahne indeksine sahip sahneye yönlendiriyor. Diğerine tıkladığımızda uygulamayı kapatıyor.

Oyunda hazır olarak aldığım tek kod “parallax.cs” arkaplandaki bulutların farklı hızlarda hareket ederek 3 boyutlu bir efekt oluşturmayı ve arkaplanın sonsuz şekilde hareket etmesini sağlıyor.