Ad: Yusuf

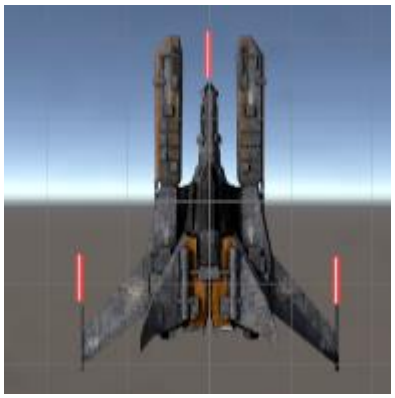
Soyad: Güney

Numara: 22360859041

Ders: Oyun Programlama Hafta 6 Rapor

1-Üçlü atış bonusu: Laser'den iki kopya daha oluşturun ve bu üç laser'in konumlarını ayarlayın.

Hierarchy penceresinde 3 adet laser nesenesi oluşturuyoruz laserlerin konumlarını, geminin ateş etme noktalarına uygun şekilde ayarlıyoruz.

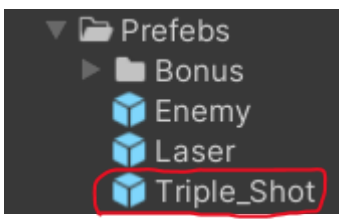


2-Boş bir oyun nesnesi oluşturun ve üç laser'i bunun child'ı olarak atayın

Hierarchy penceresinde **Create Empty** seçeneği ile oluşturduğumuz **triple_shot** adlı nesnenin içine oluşturduğumuz 3 adet laser nesnelerini yerleştiriyoruz.

3-Bu oyun nesnesinden bir prefab oluşturun

Laseri prefab yapmak için **laser** nesnesini **Assets** sekmesindeki **Prefabs** klasörüne sürükleyip bırakıyoruz



4- Üçlü bonusun hangi durumlarda aktive edileceğinin mantığını geliştirin ve kodlayın

Üçlü bonus nesnesi **Triple_Shot_Bonus** adlı nesnenin Player ile çarpışması sonucu aktif hale gelmesi gerekiyor bu işlemi ise **Bonus_sc** dosyası içerisinde aşağıdaki kod satırı ile yapıyoruz.

```
void OnTriggerEnter2D(Collider2D other){
    if(other.tag == "Player"){
        Player_sc player_sc = other.transform.GetComponent<Player_sc>();
        if(player_sc != null){
            player_sc.ActivateTripleShot();
        }
        Destroy(this.gameObject);
    }
}
```

OnTriggerEnter2D(Collider2D other): Bu metot, bir nesnenin başka bir nesneyle çarpışması durumunda çalışır.

if(other.tag == "Player") Bu satır, çarpışmanın sadece Player etiketi taşıyan nesneyle olduğunda devam etmesini sağlar. Aksi halde Bonus nesnesi **enemy** ile de çarpışabilir.

Player_sc player_sc = other.transform.GetComponent<Player_sc>(); Bu satırla oyuncunun Player_sc script'ine erişim sağlanır. Bu script, oyuncunun üçlü bonus gibi çeşitli özelliklerini kontrol eder.

player_sc.ActivateTripleShot(); Eğer **player_sc null** değilse (yani gerçekten bir oyuncu nesnesiyle çarpışma varsa), bu satır **ActivateTripleShot()** metodunu çağırır. Bu metod, üçlü bonusun belirli bir süre boyunca aktif olmasını sağlar.

Destroy(this.gameObject); Çarpışma gerçekleştiğinde bonus nesnesinin ekrandan kaldırılmasını sağlar.

5- Üçlü bonus sprite'ı toplandığında belirli bir süre için üçlü bonus'u etkinleştirin

Bu işlemi oyuncu üçlü bonus sprite'ını topladığı zaman yapacağımız için Player_sc dosyasının içerisinde önceki soruda çağırdığımız **ActivateTripleShot()** fonksiyonunu yazıyoruz.

```
public void ActivateTripleShot(){
    isTripleShotActive = true;

    StartCoroutine(TripleShotBonusDisableRoutine());
}

1 reference
IEnumerator TripleShotBonusDisableRoutine(){

    yield return new WaitForSeconds(5.0f);

    isTripleShotActive = false;
}
```

isTripleShotActive = true; satırı ile **isTripleShotActive** değişkeni **true** yapılıyor. Bu değişken üçlü bonusun aktif olup olmadığını belirlemek için kullanılır.

StartCoroutine(TripleShotBonusDisableRoutine()); satırı **TripleShotBonusDisableRoutine()** adlı bir *Coroutine*'i başlatır.

yield return new WaitForSeconds(5.0f); satırı, Coroutine'in 5 saniye boyunca duraklamasını sağlar. Yani üçlü atışın 5 saniye boyunca aktif olmasını sağlar.

5 saniye sonra **isTripleShotActive** değeri **false** yapılarak üçlü atış bonusu devre dışı bırakılır.

6-FireLaser fonksiyonunda gerekli güncellemeleri yapın

Player_sc dosyası içerisindeki shoting() fonksiyonu içerisinde aşağıdaki değişiklikleri yapıyoruz.

```
void shoting(){
if(Input.GetKeyDown(KeyCode.Space) /*&& Time.time > nextFire*/){

    if(!isTripleShotActive)
    {
        Instantiate(laserPrefab, transform.position + new Vector3(0,1.05f,0),Quaternion.identity);
    }
    else
    {
        Instantiate(tripleShotPrefab, transform.position + new Vector3(-0.9f,1.05f,0),Quaternion.identity);
    }

    //nextFire = Time.time + fireRate;

    }
}
```

if (!isTripleShotActive) Bu koşul **isTripleShotActive** değişkeni **false** ise çalışır Yani üçlü bonus aktif değilse ek atış yapılır.

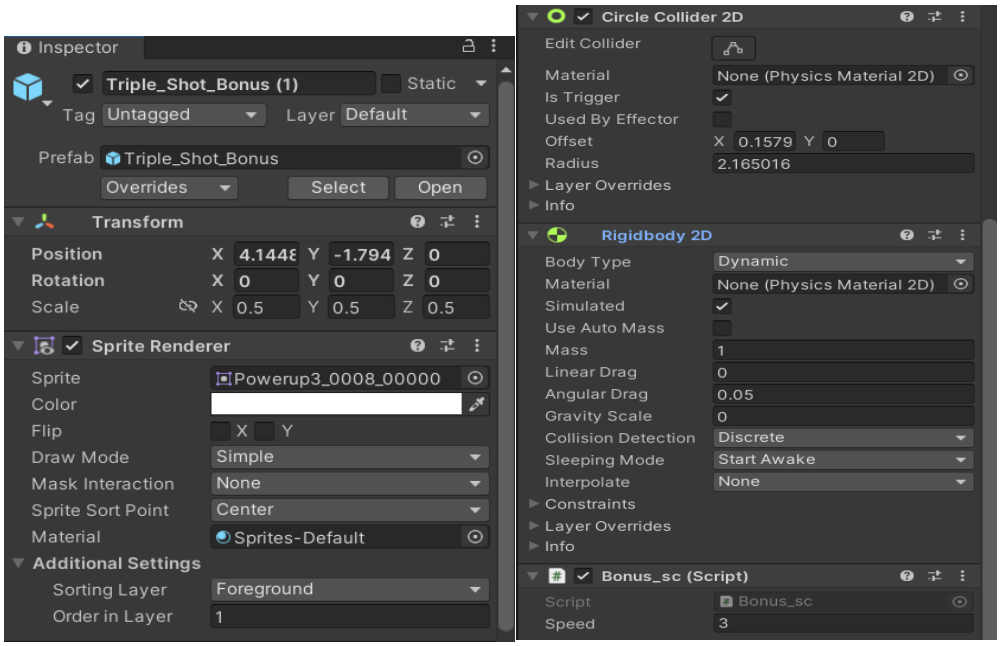
Eğer **isTripleShotActive true** ise yani üçlü bonus aktifken else bloğu çalışır.

Instantiate(tripleShotPrefab, transform.position + new Vector3(-0.9f, 1.05f, 0), Quaternion.identity); Bu satır üçlü bonusun aktif olduğu durumda çağrılır. **tripleShotPrefab** adlı üçlü lazer nesnesi, karakterin belirli bir konumundan ateşlenir. **tripleShotPrefab** 3 adet laser içerir, böylece oyuncu üçlü atış yapmış olur.

7-Üçlü bonus sprite'ından bir nesne oluşturun, ölçeğini ayarlayın, circle collider ve rigidbody ekleyin, sorting layer'ı düzenleyin

Assets sekmesinin altındaki **Sprites** klasörünün içerisindeki **Power_Ups** dosyasının içerisinde **Triple_Shot** klasörü bulunmakta bu klasörün içerisindeki ilk nesneyi **Hierarchy** penceresine sürükleyerek üçlü bonus sprite'ını sahneye ekliyoruz.

Daha sonra ismini **Triple_Shot_Bonus** olarak değiştiriyoruz. **Inspecter** penceresindeki **Transform** kısmından nesnenin ölçeğini ayarlıyoruz. Cisme **Add Component** seçeneğiyle **Circle Collider 2D**, **Rigidbody 2D**'yi ekliyoruz. Burada Üçlü atış bonusu Player'a çarptığında kaybolacağı için **is Trigger**'ı aktif hâle getiriyoruz. Son olarak **Sprite Renderer** kısmından sorting layerı Foreground olarak ayarlıyoruz.



8-Script oluşturarak üçlü bonus nesnesi ile ilişkilendirin, script ile nesnenin belirli bir hızla hareket etmesini sağlayın. Ekran dışına çıkıldığında nesneyi yok edin. Çarpışma kontrolü ekleyin.

Script dosyasının içerisine **Bonus_sc** adlı bir C# dosyası oluşturarak bu dosyayı **Triple_Shot_Bonus** nesnesinin üzerine taşıyoruz. Daha sonra üçlü bonusun hareketini sağlayan aşağıdaki kod satırlarını **Bonus_sc** dosyasının içerisine yazıyoruz.

```
void Update()
{
    transform.Translate(Vector3.down * speed * Time.deltaTime);

     if(transform.position.y < -5.8f){
        Destroy(this.gameObject);
    }
 }
```

Bu kod satırı cismin y ekseninde aşağı yönde hareketini sağlar eğer cisim sahne dışına çıkarsa (y ekseninde -5.8f pozisyonu) nesneyi yok eder.

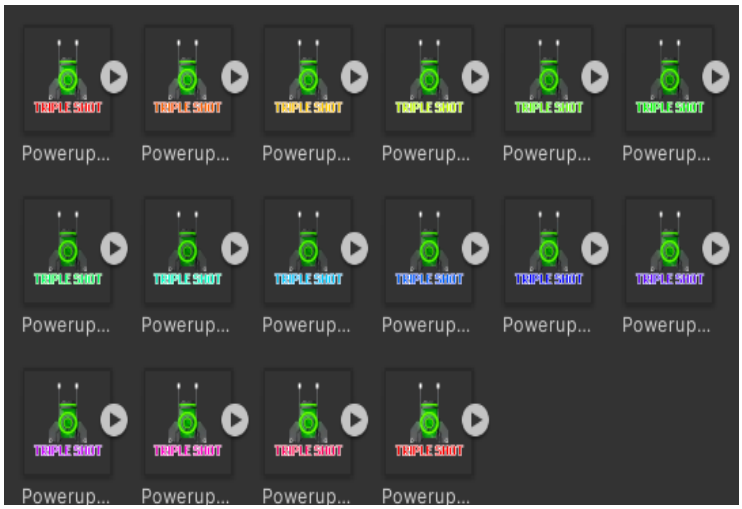
9-Player kodu içinden üçlü bonus koduna erişin ve üçlü bonus'u etkinleştirin

```
void OnTriggerEnter2D(Collider2D other){
   if(other.tag == "Player"){
      Player_sc player_sc = other.transform.GetComponent<Player_sc>();
      if(player_sc != null){
          player_sc.ActivateTripleShot();
      }
      Destroy(this.gameObject);
   }
}
```

İşaretli kod satırında player_sc dosyası içerisindeki ActivateTripleShot() fonksiyonuna erişerek üçlü atışı aktif hâle getiriyoruz.

10-Üçlü atış bonusu nesnesi için animasyon ekleyin

Proje ana sayfasında, sol üst köşede bulunan **Window** menüsünden **Animation** seçeneğini seçerek animasyon penceresini ekliyoruz. Ardından, **Assets** klasörünün altında **Animation** adında yeni bir klasör oluşturuyoruz. Bu klasörde, *Create* seçeneğine tıklayıp "**Triple_Shot_Bonus.anim**" adında bir animasyon dosyası yaratıyoruz. Sonrasında, *Triple_Shot_Bonus* nesnesini seçip, ***Sprites*** klasöründeki triple shot ile ilgili sprite'ları, **Animation** penceresinin sol tarafına sürükleyerek animasyonu oluşturuyoruz.

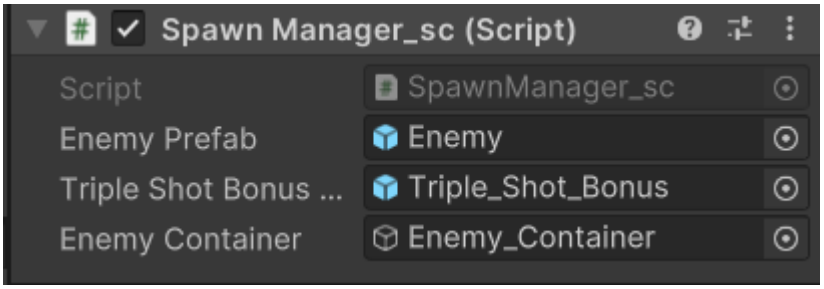


11-SpawnManager'ı üçlü atış bonusunu da dikkate alacak şekilde düzenleyin

```
1 reference
IEnumerator SpawnBonusRoutine(){
    while(stopSpawning == false){

        Vector3 position = new Vector3(Random.Range(-9.4f, 9.4f), 7.4f, 0);
        Instantiate(tripleShotBonusPrefab, position, Quaternion.identity);
        yield return new WaitForSeconds(20.0f); // bonusun gelme süresi
    }
}
```

Bu fonksiyon da 20 saniyede bir x ekseninde rastgele bir noktada bir adet üçlü bonus oluşturulmasını sağlıyor. Bu fonksiyonu çalıştırma işlemini Start() fonksiyonunu içerisinde gerçekleştiriyoruz.



Triple_Shot_Bonus değişkenine de Triple_Shot_Bonus prefebini atayarak oyunu çalıştırıyoruz.

Kodların tamamına Github sayfası üzerinden ulaşabilirsiniz.

Github Linki: