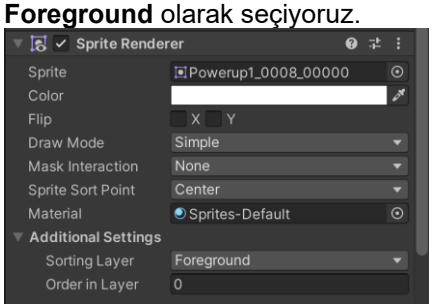
Oyun Programlama Ödevi – Hafta 7

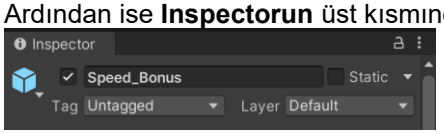
Ad: Eren
Soyad: Köse
Numara: 22360859075
GitHub Kod Linki: https://github.com/erennkose/btu-oyun-programlama/tree/main/Hafta7

- ## Hız bonusu: Hız sprite'ını kullanarak bir prefabrik oluşturun (sorting layer, isim, ölçek, collider, rigid body, script ekle/düzenle)

Assets>Sprites>Power Ups>Shield klasörü altından ilk sıradaki spriteı sürükle-bırak yöntemiyle **Hierarchy** kısmına bırakıyoruz. Ardından **Inspector** kısmından **Sprite Renderer** bölümü altında **Sorting Layer**'ımızı **Foreground** olarak seçiyoruz.

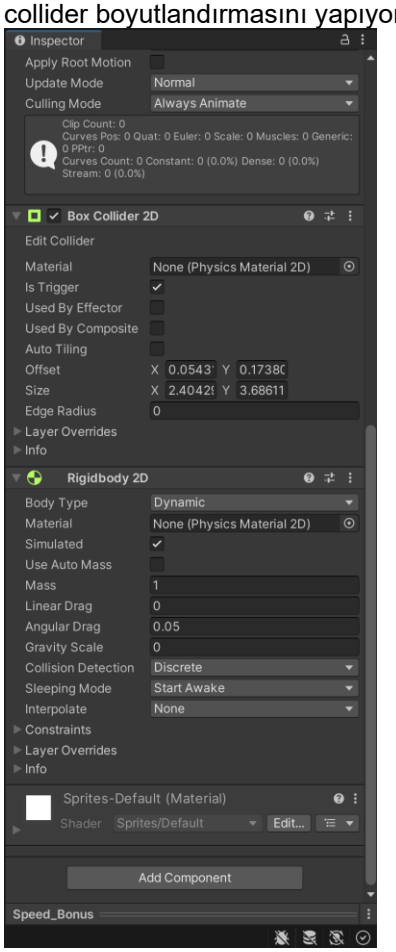


Ardından ise **Inspectorun** üst kısmından bu nesnemizin adını **Speed_Bonus** yapıyoruz.



Inspector kısmındaki **Transform** bölümü altındaki **Scale** özelliklerini ayarlayarak boyutumuzu ayarlıyoruz.

Ardından **Inspector** kısmında en alta inip **Add Component** butonuna tıklıyoruz. Burada **Box Collider 2D** ve **Rigidbody 2D** özelliklerini ekliyoruz. Collider için **Edit Collider** butonuna tıklayıp **Scene** kısmından gerekli collider boyutlandırmasını yapıyoruz. **Rigidbody** için ise **Gravity Scale'ımızı 0 yapıyoruz**.



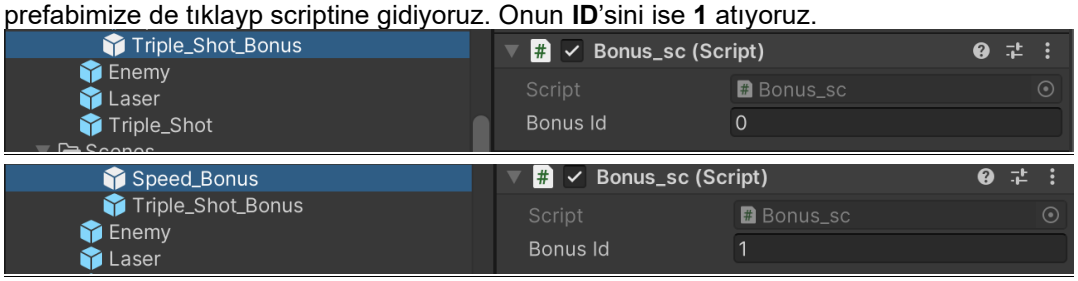
Bu işlemlerin ardından Bonus_sc scriptimizi sürükle-bırak ile bu nesnemizin üzerine bırakıyoruz. Ardından scriptimize tıklayıp kod editörümüzde kodumuzu açıyoruz. Bonuslarımız artık birden fazla olduğundan dolayı bu scriptte bazı düzenlemeler yapmamız gerekecek. Bu düzenlemelerden önce ise Unity uygulamamıza dönüyoruz ve sürükle-bırak yöntemiyle **Speed_Bonus** nesnemizi **Prefabs** klaörü altındaki **Bonus** klasörüne bırakıyoruz. Bu sayede nesnemizi prefabe dönüştürmüş oluyoruz. Şimdi koda geri dönebilriz.

- ## Bonus script'ini modüler hale getirin (bonus'ları tekil olarak numaralandırın)

Bonuslarımızı ayırt edebilmemiz için onlara **ID** ataması yapacağız. Bunun için öncelikle **Bonus_sc** içerisinde **public int bonusId** değişkeni tanımlıyoruz.

```
public int bonusId;
```

Bu tanımlamanın ardından Unity uygulamamıza gidiyoruz ve **Triple_Shot_Bonus** prefabimize tıklayıp **Inspector** kısmında scriptimizi bulup altındaki **Bonus ID** değerini **0** atıyoruz. Aynı şekilde **Speed_Bonus** prefabimize de tıklayp scriptine gidiyoruz. Onun **ID**'sini ise **1** atıyoruz.



- ## Tekil bonus isine göre bonus'u etkinleştirin

Ardından ise kodumuza geri dönüyoruz ve **OnTriggerEnter2D** fonksiyonumuzda bazı değişiklikler yapıyoruz. Burada **playersc** değişkenimizin **null olmadığı** durumun içerisine hangi bonusu aldığımızı tespit edecek bir **switch-case** yapısı yazıyoruz.

```
void OnTriggerEnter2D(Collider2D other){

    if(other.tag == "Player"){
        Player_sc playersc = other.transform.GetComponent<Player_sc>();
        if(playersc != null){
            switch (bonusId){
                case 0:
                playersc.ActivateTripleShot();
                break;
                case 1:
                playersc.ActivateSpeedBonus();
                break;
                case 2:
                playersc.ActivateShieldBonus();
                break;
                default:
                Debug.Log("Default value in switch case");
                break;
            }
        }
        Destroy(this.gameObject);
    }
}
```

Bu sayede **temas ettiğimiz bonusa göre** tepki vermemiz sağlanmış oluyor.

- ## Player script'i içinde hız çarpanı değişkeni ve hız bonus'unun aktif olup olmadığı ile ilgili değişkeni tanımlayın

Bu işlem için ise Unity uygulamamızdan Player_sc scriptimizi açıyoruz. Hız bonusumuzun hızımızı ikiye katlamasını sağlamak için **speedMultiplier** adında bir **float** değişken tanımlıyoruz ve değerini **2** yapıyoruz.

```
float speedMultiplier = 2;
```

Değişkenimizi tanımladıktan sonra bir de **isSpeedBonusActive** adında bir **bool** değişken tanımlayıp değerini **false** giriyoruz.

```
public bool isSpeedBonusActive = false;
```

- ## Hız bonus'unu aktifleştirmek için fonksiyon tanımlayın

Hızla ilgili değişkenlerimizi de tanımladıktan sonra hız bonusumuzu aktifleştirmek için fonksiyon tanımlayacağız. Player_sc altında **ActivateTripleShot fonksiyonuna benzer** bir fonksiyon tanımlayacağız. Bu fonksiyon **isSpeedBonusActive bool** değişkenimizi **true** yapacak ve **speed** değişkenimizi **speedMultiplier** değişkenimizle **çarpacak**. Ardından ise bu bonusun sınırsız aktifliğini engellemek için bir **coroutine fonksiyonu çağıracak**.

```
public void ActivateSpeedBonus(){
    isSpeedBonusActive = true;
    speed *= speedMultiplier;
    StartCoroutine(SpeedBonusDisableRoutine());
}
```

- ## Hız bonus'unu pasif hale getirecek routine tanımlayın

Hız bonusumuzun sınırsız aktifliğini engellemek için gereken coroutine fonksiyonumuzu ise **TripleShotDisableRoutine fonksiyonuna benzer** olacak ve bonustan 5 saniye sonrasında **hızı speedMultiplier** ile **bölecek** ve **isSpeedBonusActive** değişkenini **false** yapacak. Bu bir coroutine fonksiyon olduğundan dolayı da **IEnumerator** olarak tanımlanmış bir fonksiyon olacak.

```
IEnumerator SpeedBonusDisableRoutine(){
    yield return new WaitForSeconds(5.0f);
    speed /= speedMultiplier;
    isSpeedBonusActive = false;
}
```

- ## Player kodu içinden hız bonusu koduna erişin ve hız bonus'unu (gerektiğinde) etkinleştirin

Bunun için ise **Bonus_sc** scriptimizdeki **OnTriggerEnter2D** fonksiyonumuza geri dönüyoruz. Burada **Player_sc playersc = other.transform.GetComponent<Player_sc>();** kod parçasıyla zaten Player_sc scriptimizdeki fonksiyonlara ulaşabiliyorduk. Bu **playersc** değişkenimizi kullanarak switchimiz altındaki **case 1** durumunda speed bonusumuzu aktifleştirecek olan **ActivateSpeedBonus** fonksiyonunu çağırıyoruz. Böylece gerektiği durumda Player kodumuz içindeki hız bonusuna erişip aktifleştirmiş oluyoruz.

```
void OnTriggerEnter2D(Collider2D other){

    if(other.tag == "Player"){
        Player_sc playersc = other.transform.GetComponent<Player_sc>();
        if(playersc != null){
            switch (bonusId){
                case 0:
                playersc.ActivateTripleShot();
                break;
                case 1:
                playersc.ActivateSpeedBonus();
                break;
                case 2:
                playersc.ActivateShieldBonus();
                break;
                default:
                Debug.Log("Default value in switch case");
                break;
            }
        }
        Destroy(this.gameObject);
    }
}
```

- ## SpawnManager'ı düzenleyerek belirli aralıklarla hız bonusu üretin

SpawnManager_sc scriptimizi açıyoruz. Burada bulunan **SpawnBonusRoutine** fonksiyonu üzerinde bazı değişiklikler yapacağız. Öncelikle fonksiyonların dışında **tripleShotBonusPrefab** adındaki prefabimiz yerine bir **GameObject dizisi** tanımlıyoruz.

```
GameObject[] bonusPrefabs;
```

Ardından **SpawnBonusRoutine** fonksiyonumuza gidiyoruz ve içerisindeki **while** döngüsünün **içinde** bonuslarımızdan rastgele şekilde oluşturulmasını sağlamak için **rastgele 0 ile 3 arasında** (3 dahil değil) belirlenen bir değeri integer değişkene atıyoruz.

```
int randNum = Random.Range(0,3);
```

Ardından **Instantiate** fonksiyonumuzun parametrelerini şu şekilde düzenliyoruz;

```
Instantiate(bonusPrefabs[randNum], position, Quaternion.identity);
```

Bu değişiklikler sayesinde rastgele bir şekilde 10 saniye aralıklarla herhangi bir bonusu spawnlamış oluyoruz. (3 olmasının sebebi Shield bonusunun da gelecekte eklenecek olması)

```
IEnumerator SpawnBonusRoutine(){
    while(stopSpawning == false)
    {
        int randNum = Random.Range(0,3);
        Vector3 position = new Vector3(Random.Range(9.4f,-9.4f), 7.4f, 0);
        Instantiate(bonusPrefabs[randNum], position, Quaternion.identity);
        yield return new WaitForSeconds(10.0f);
    }
}
```

- **Hız bonusu için animasyon ekleyin**

Bunun için öncelikle Unity uygulamamıza geri dönüyoruz. Ardından prefab hale çevirdiğimiz **Speed_Bonus** prefabini sürükle bırak ile **Hierarchydeki** boşluğa bırakıyoruz. Ardından bu nesnemiz üzerine basılı iken **Animation** sekmesine geçiyoruz. **Animation** sekmesinde **Create** diyoruz ve açılan pop-uptan **Animations klasörümüzü** seçip ismini **Speed_Bonus_anim** koyuyoruz. Bu işlemin ardından **Assets>Sprites>Power Ups>Speed** klasörümüzün altındaki tüm görselleri **shift** tuşu ile seçiyoruz ve sürükleyip **Animations** sekmemize bırakıyoruz. Bu işlemin ardından **Speed_Bonus** nesnemize tıklıyoruz ve üst kısımda bulunan **Overrides** kısmına tıklayıp Prefabimizin de animasyona sahip olması için Override işlemini onaylıyoruz. Bu işlemin ardından **Speed_Bonus** nesnemizi silebiliriz. Bu işlemler sonucunda da Speed_Bonus prefabimize animasyon eklenmiş oluyor.