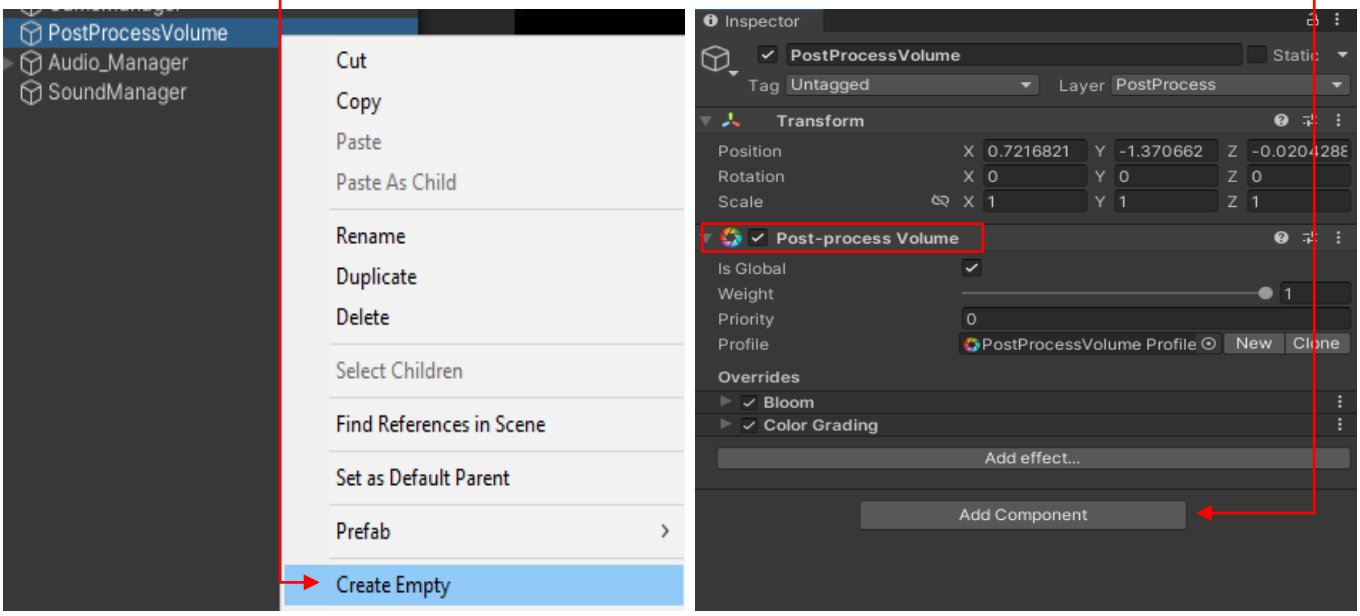
11. Hafta Raporu – Aleks Dulda 21360859025

Post processing paketinin kurulumu:

1. Unity Projenizi Açın: Unity Editor'ü açın ve üzerinde çalıştığınız projeyi yükleyin.
2. Package Manager'i Açın:
   - Menü çubuğundan Window > Package Manager seçeneğine tıklayın.
3. Post Processing Paketini Bulun:
   - Unity Registry bölümünde "Post Processing" yazın. Eğer bu sekmeyi göremiyorsanız, sol üst köşeden "Packages: Unity Registry" seçeneğini seçin.
4. Paketi Yükleyin:
   - "Post Processing" paketini seçin ve sağ alt köşede bulunan Install düğmesine tıklayın.

Projeye post processing uygulanması:

1. Post Processing Volume için GameObject Ekleyin:
   - Scene Hierarchy'ye sağ tıklayın ve Create Empty seçeneğini seçerek bir boş GameObject oluşturun.
   - Bu GameObject'e anlamlı bir isim verin (örneğin, "PostProcessingVolume").
2. Post-Processing Volume Bileşenini Ekleyin:
   - Bu GameObject'i seçin ve Inspector penceresinde Add Component butonuna tıklayın.
   - "Post-Processing Volume" bileşenini ekleyin.
3. Global Olarak Ayarlayın:
   - Post-Processing Volume bileşeninde Is Global seçeneğini işaretleyin.

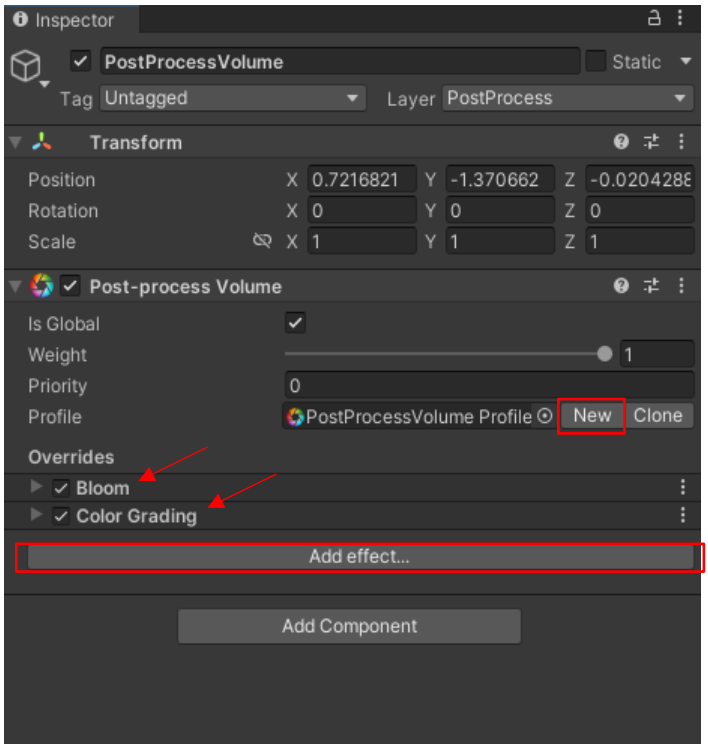


4. **Yeni Bir Profil Oluşturun:**

   - Post-Processing Volume bileşeninde **New Profile** butonuna tıklayın.
   - Bu işlem yeni bir profil oluşturacak ve düzenlemenize izin verecektir.

5. **Efektleri Ekleyin:**

   - Profili düzenlemek için açılan pencerenin sağındaki **Add Override** butonuna tıklayın.
   - **Unity > Post-Processing** kategorisinden efektleri seçebilirsiniz.
   - Örnek olarak:
     - **Bloom**: Sahneye parlaklık efekti ekler.
     - **Color Grading**: Renk, ton ve kontrast üzerinde kontrol sağlar.

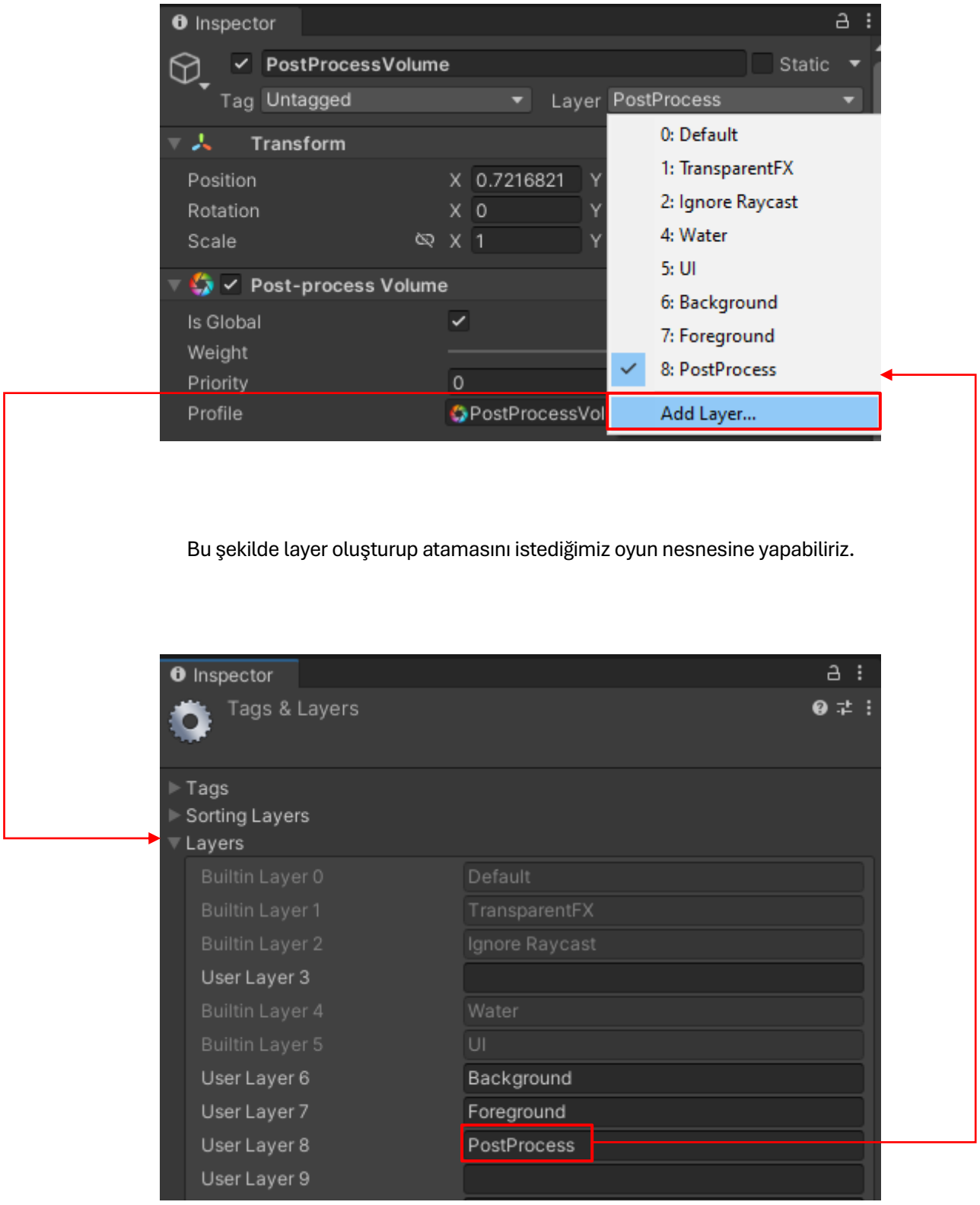


6. **Kameraya Post-Processing Layer Ekleme:**

   - **Main Camera**'yı seçin.
   - **Inspector** penceresinde **Add Component** butonuna tıklayın ve "Post-Processing Layer" ekleyin.

7. **Layer Ayarlarını Yapın:**

   - **Layer** seçeneğini "PostProcessing" olarak ayarlayın (yoksa yeni bir **PostProcessing** layer'ı oluşturun).
   - Bu layer, Post-Processing Volume'un etkisini görmesini sağlar.

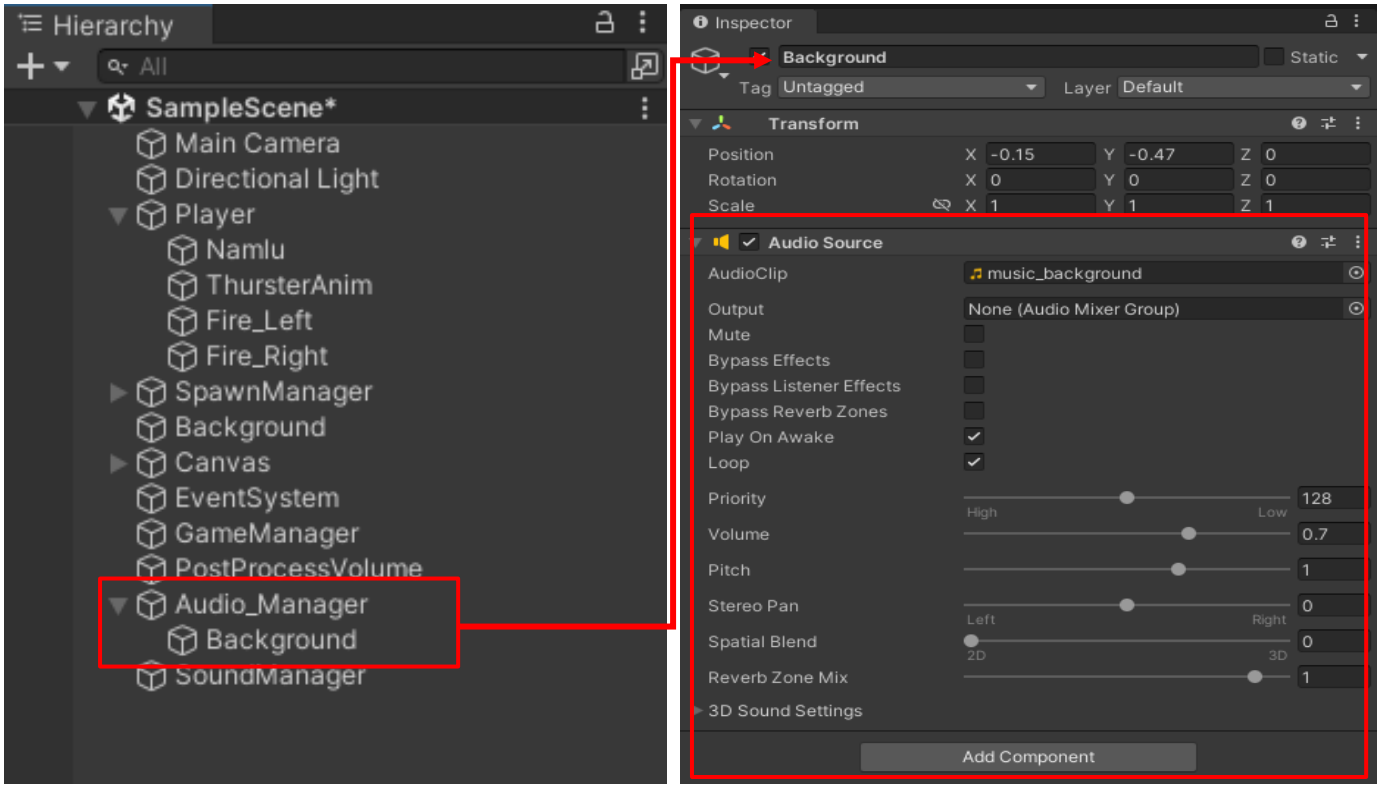


Bu şekilde layer oluşturup atamasını istediğimiz oyun nesnesine yapabiliriz.

Audio Ekleme

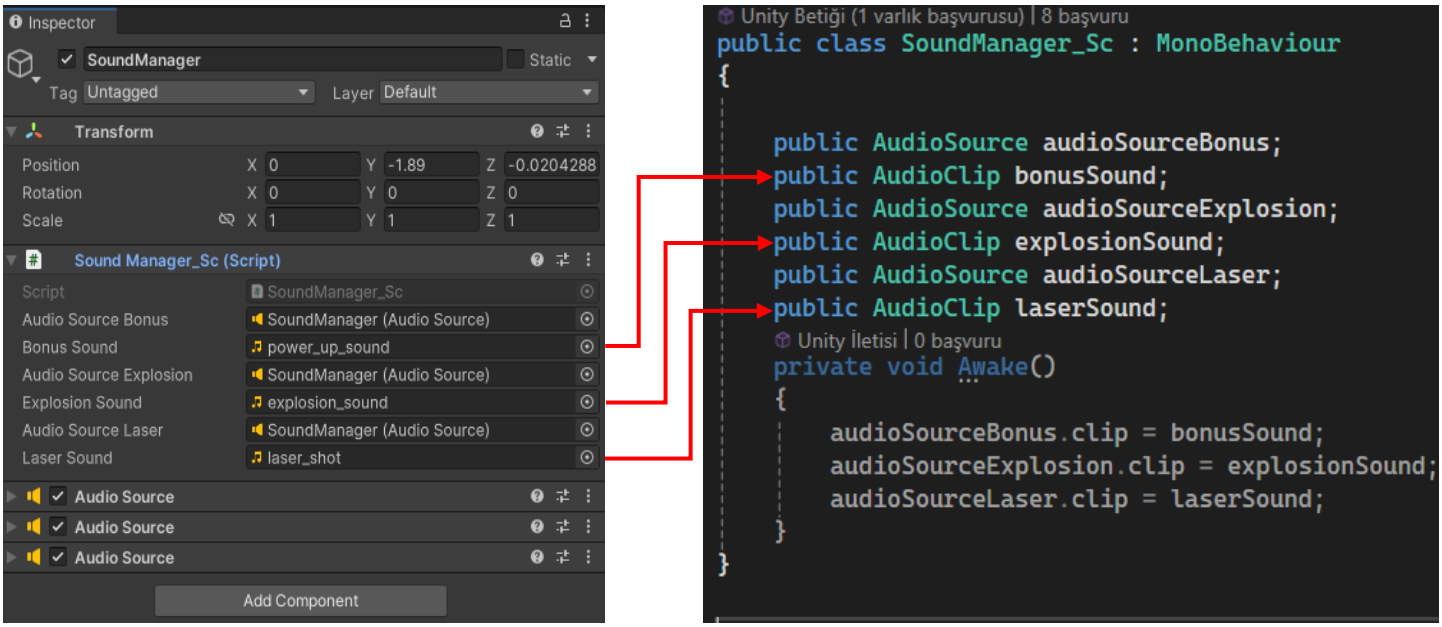
1. Arka Plana Ses Eklenmesi

- Boş bir GameObject oluşturulup, ismi "Audio_Manager" olarak belirlendi.
- "Audio_Manager" nesnesinin altına bir boş GameObject daha oluşturulup, ismi "Background" olarak belirlendi.
- "Background" nesnesine AudioSource component'i eklendi.
- Play On Awake seçeneği işaretlendi, böylece oyun başladığında müzik otomatik olarak çalmaya başlar.
- Loop seçeneği işaretlendi, böylece müzik sürekli olarak tekrar eder.
- AudioClip kısmına var olan bir arka plan müziği eklendi.



2. Ses Efektleri İçin SoundManager

- Patlama, Lazer ve PowerUp ses efektleri için bir boş GameObject oluşturulup, ismi "SoundManager" olarak belirlendi.
- **3 adet AudioSource component'i** eklendi, her bir ses efekti için yeterli ses çıkışı sağlanması amaçlandı (bu şekilde bir efekt çalarken diğerleri engellenmez).
- **SoundManager** nesnesine "SoundManager_Sc" adında bir script dosyası eklendi.
- Script dosyasına gerekli kodlar yazıldı.



- Script içerisinde her bir ses efekti için bir AudioSource ve bir AudioClip tanımladım. Awake() fonksiyonu içinde, her bir AudioSource'a ilgili AudioClip'i atıyorum. Bu sayede, oyun başladığında her ses efekti doğru şekilde hazır oluyor ve çalmak için ihtiyaç duyduğumda hemen kullanılabiliyor.

1. **Değişken Tanımlaması:**
    - **Astreoid_Sc** script dosyasına, **SoundManager_Sc** türünde bir değişken ekliyorum:
    - SoundManager_Sc soundManager;
    - Bu, **SoundManager_Sc** türünde bir değişken oluşturur.
2. **SoundManager'a Erişim Sağlama:**
    - **soundManager** değişkenine, **GameObject.FindObjectOfType<SoundManager_Sc>()** kodunu ekleyerek **SoundManager_Sc** script'ine erişim sağlıyorum:
    - soundManager = GameObject.FindObjectOfType<SoundManager_Sc>();
    - Bu sayede, **SoundManager_Sc** script'ini bulmuş ve ona referans vermiş oluyoruz.
3. **Ses Efekti Çalma:**
    - Son olarak, patlama sesini çalmak için şu kodu kullanıyorum:
    - soundManager.audioSourceExplosion.Play();
    - Bu kodu, asteroid yok olmadan önce çağırarak patlama sesini çalıştırıyorum.

Bu örnekte olduğu gibi diğer ses efektlerini de ekleyebilir oyunu daha zevkli hale getirebiliriz.

```
[SerializeField]
SoundManager_Sc soundManager;


// Start is called before the first frame update
Unity İletisi | 0 başvuru
void Start()
{
    soundManager = GameObject.FindObjectOfType<SoundManager_Sc>();
    if (soundManager == null)
    {
        Debug.LogError("SoundManager_Sc not found!");
    }
```

```
Unity İletisi | 0 başvuru
void OnTriggerEnter2D(Collider2D other) // temasa giren
{
    if (other.tag == "Player")
    {
        if (player.isShieldActive)
        {
            player.isShieldActive = false;
        }
        else
        {
            player.Damage();
        }
        soundManager.audioSourceExplosion.Play();
        anim.SetTrigger("IsTouch");
        speed = 0;
        Destroy(gameObject, 1.0f); // enemy'i yok et
    }
    else if (other.tag == "Lazer")
    {
        Destroy(other.gameObject); // lazeri yok et
        anim.SetTrigger("IsTouch");
        speed = 0;
        soundManager.audioSourceExplosion.Play();
        Destroy(gameObject, 1.0f); // enemy'i yok et
        player.score += 10;
    }
}
```

Kaynakça

https://en.wikipedia.org/wiki/Unity_(game_engine)

https://chatgpt.com/

https://docs.unity3d.com/Manual/index.html

Github link

https://github.com/AleksDulda/GamePrograming

https://github.com/AleksDulda/GamePrograming/tree/main/Rapor/Week_11