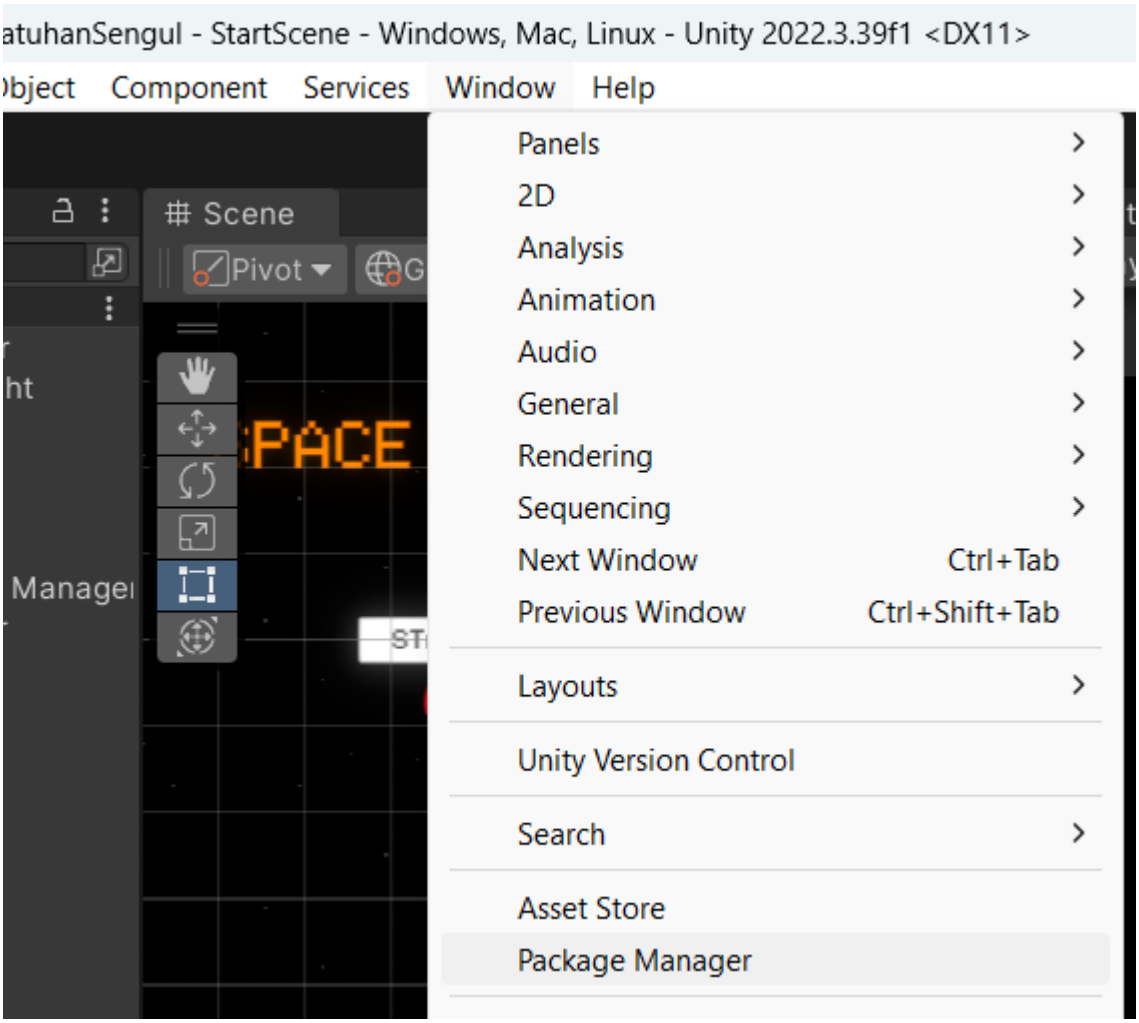
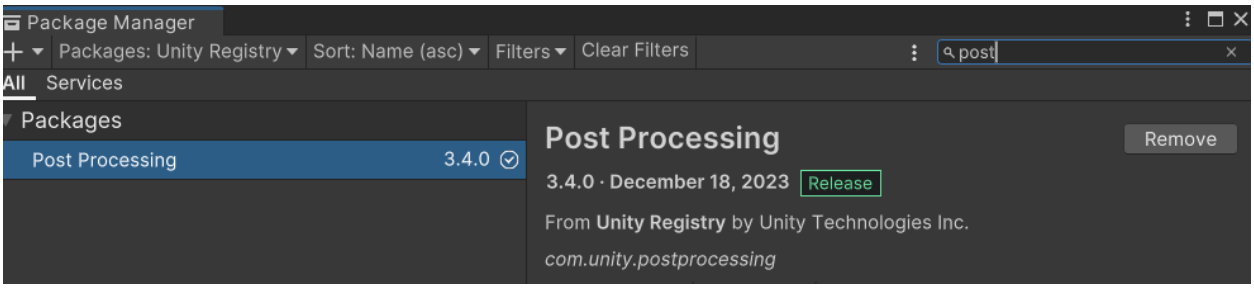
10.Hafta Ders Raporu – Batuhan Şengül

Post Processing

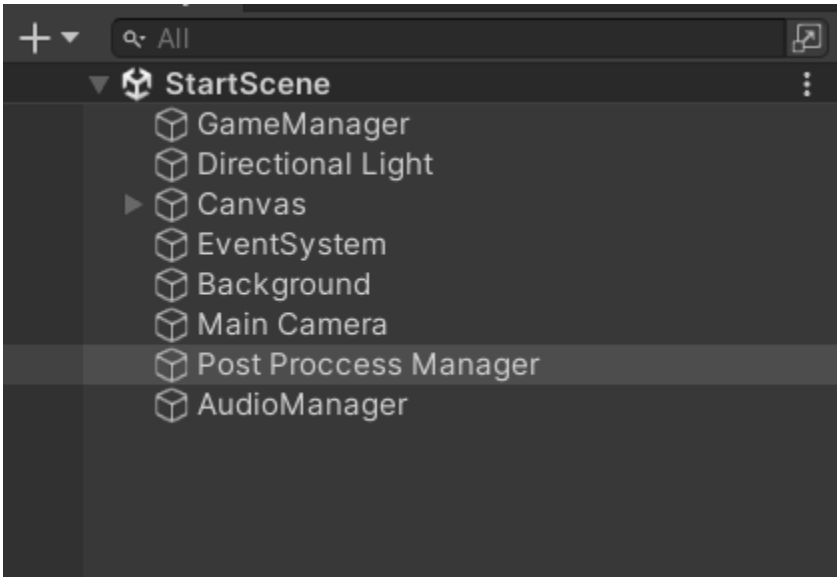
Projeye post proceesing eklemek için.



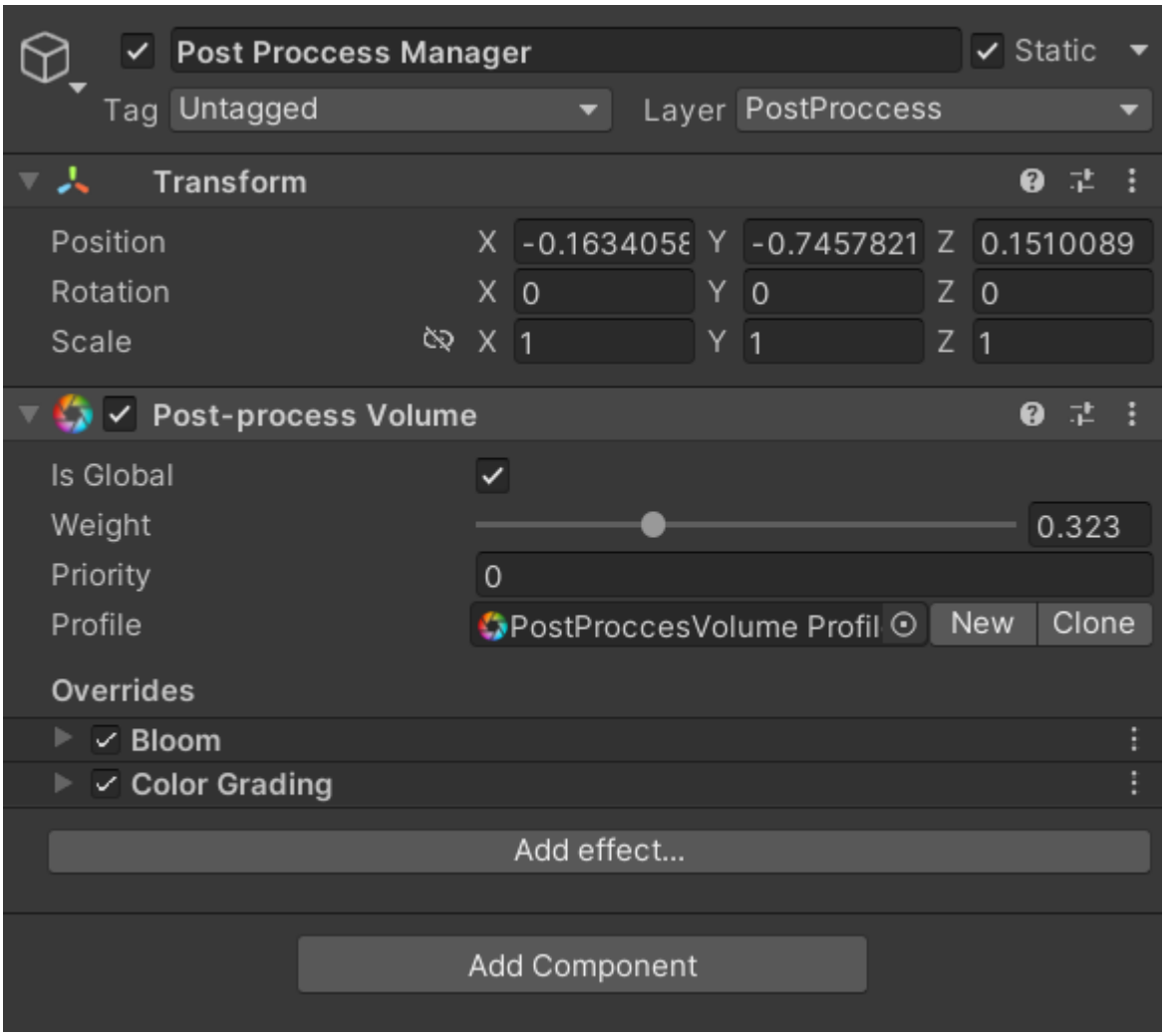
Window -> package manager seçilir.



Packages sekmesinde unity registiry seçilir. Arama kısmına istenilen paketin adı yazılır ve install tuşuna basılarak projeye dahil edilir.



Yeni boş bir oyun nesnesi oluşturulur ve adı “Post Proccess Manager” olarak ayarlanır.

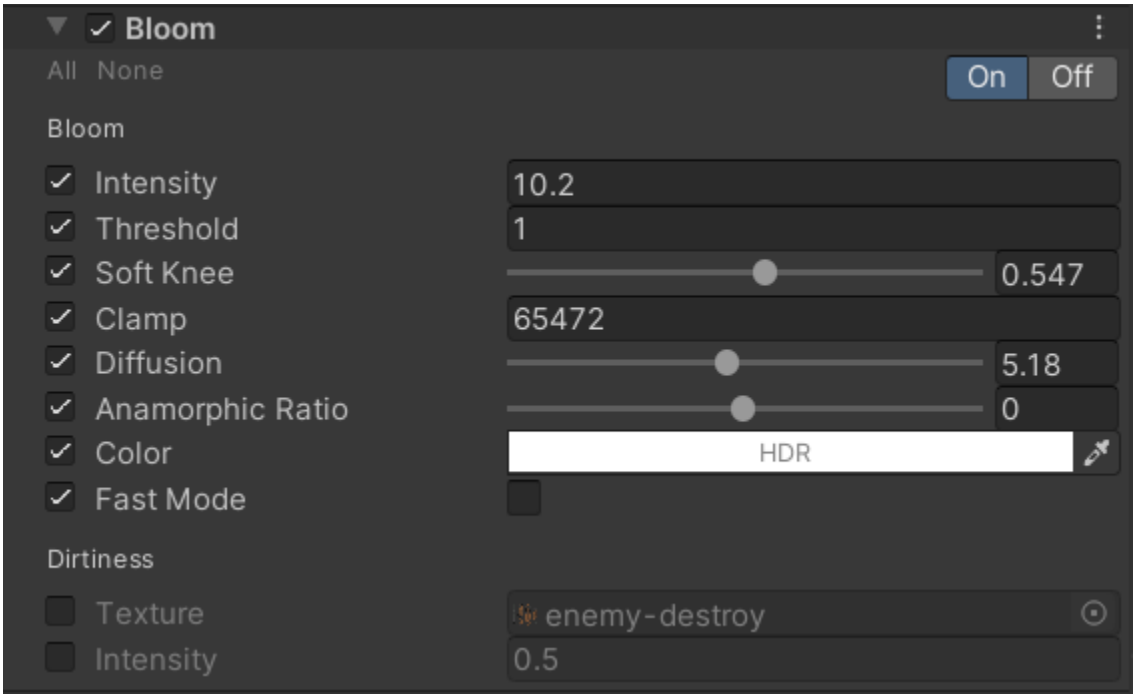


Oluşturulan oyun nesnesine Post-Process Volume bileşeni eklenir. Layer'ı yeni oluşturulan "PostProcess" olarak ayarlanır.

Add effect tuşunua basılarak istenilen efektler eklenebilir. Bu örnektel Bloom ve Color Grading efektleri eklenmiştir.

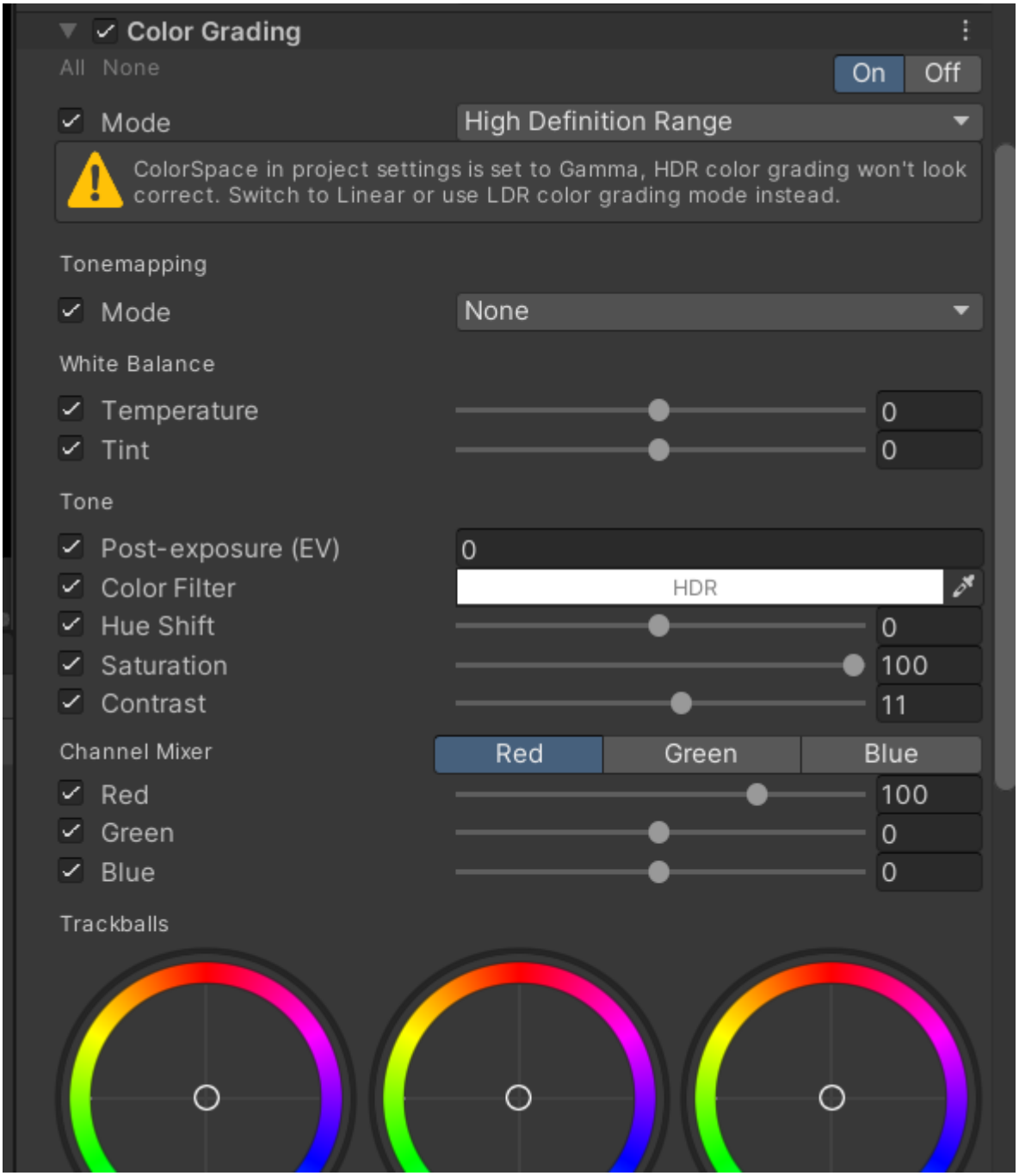
Eklenen efektlerin detayları isteğe bağlı olarak ayarlanır.

Bloom



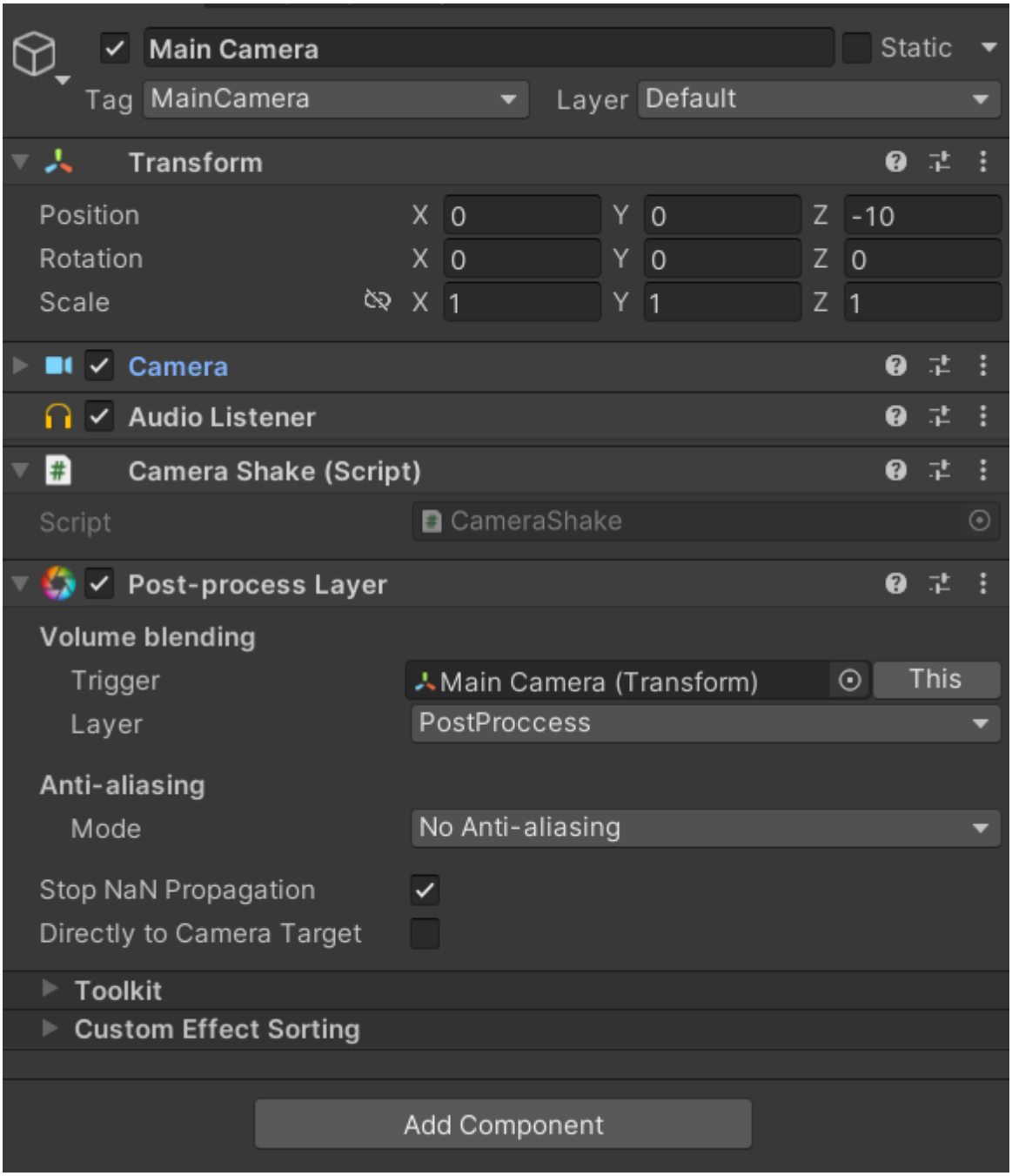
Oyun için bu şekilde düzenlenmiştir.

Color Grading



Oyun için bu şekilde düzenlenmiştir.

Main Camera



Oyunun görüntü aldığı kameranın bu yapılan efektlerden etkilenmesi için main camera oyun nesnesine; Post-Process Layer bileşeni eklenir Layer ayarları düzenlenir ve efektlerimiz oyuz üzerinde etki alır.

Post-Process efektlerinden sonra oyun

Post-Process efektleri olmadan oyun

Sesler

Sesler için iki tane script oluşturulmuştur biri ses sınıfının genel özelliklerini tutan bir Sound.cs base sınıfı diğeri de seslerin genel olarak oynatılmasını durdurulmasını vb. durumları kontrol eden bir Singleton yapısında bir AudioManager.cs scripti.

Sound.cs

```
[System.Serializable]

public class Sound
{
    public string name;

    public AudioClip clip;

    [Range(0f, 1f)]
    public float volume = .75f;
    [Range(0f, 1f)]
    public float volumeVariance = .1f;

    [Range(.1f, 3f)]
    public float pitch = 1f;
    [Range(0f, 1f)]
    public float pitchVariance = .1f;

    public bool loop = false;
    public bool playOnAwake = false;

    public AudioMixerGroup mixerGroup;

    public bool onDifferentGO=false;
    [HideInInspector] public GameObject attachedObject;


    [HideInInspector]
    public bool attachedOnPlayer = false;
    [HideInInspector]
    public AudioSource source;

    [HideInInspector]
    public float minDistance=1f, maxDistance=500f, spatialBlend=0f;
```

Bir ses nesnesinin basit özelliklerini tutan sınıf. Detaylı olarak açıklamayı düşünülmemektedir çünkü işin inekliğini yaptım ve uzun vadede kullanımıma uyun bir yapı oluşturdum.

AudioManager.cs

- **Değişkenler**

```
public static AudioManager Instance;

public AudioMixerGroup mainMixerGroup;

public Sound[] sounds;
public float fadeInTime;
public float fadeOutTime;

private List<Sound> unassignedPlayerSounds = new List<Sound>();
```

- **Awake() Fonksiyonu**

```
void Awake()
{
    if (Instance == null)
    {
        Instance = this;
    }
    else
    {
        Destroy(gameObject);
        return;
    }

    DontDestroyOnLoad(gameObject);

    foreach (Sound s in sounds)
    {
        if (s.attachedOnPlayer)
        {
            unassignedPlayerSounds.Add(s);
        }
        else
        {
            AttachSoundToObject(s, s.attachedObject != null ? s.attachedObject : gameObject);
        }
    }
}
```

Singelton yapısı kurulur. Ses yöneticisindeki tüm sesler oluşturulur, hierchyde gerkli yerlere eklenirler.

- **Start() Fonksiyonu**

```
Unity Message | 0 references
private void Start()
{
    Play("MainTheme");
    SceneManager.sceneLoaded += OnSceneLoaded;
}
```

Oyunun ana müziği başlatılır ve sahne yüklemelerini izleyen bir dinleyici atanır.

- **OnDestroy() Fonksiyonu**

```
Unity Message | 0 references
private void OnDestroy()
{
    SceneManager.sceneLoaded -= OnSceneLoaded;
}
```

Başlangıçta oluşturulan dinleyici nesnenin yok olmasına karşın kaldırılır. (Memory leak olasın diye)

- **OnPlayerLoaded(GameObject newPlayer) Fonksiyonu**

```
public void OnPlayerLoaded(GameObject newPlayer)
{
    foreach (Sound s in unassignedPlayerSounds.ToArray())
    {
        if (newPlayer != null)
        {
            AttachSoundToObject(s, newPlayer);
            //unassignedPlayerSounds.Remove(s);
        }
    }
}
```

Oyuncu oyun nesnesi oluşturulduğunda çalışır oyuncunun üzerinde olacak sesler atamasını sağlar. Örneğin ateş etme, zıplama gibi oyuncunun kendisinin çıkartabileceği sesler.

- **AttachSoundToObject(Sound s, GameObject targetObject) Fonksiyonu**

```
2 references
public void AttachSoundToObject(Sound s, GameObject targetObject)
{
    if (s.source != null)
    {
        Destroy(s.source);
    }

    s.attachedObject = targetObject;
    s.source = targetObject.AddComponent<AudioSource>();


    s.source.clip = s.clip;
    s.source.volume = s.volume;
    s.source.loop = s.loop;
    s.source.playOnAwake = s.playOnAwake;
    s.source.outputAudioMixerGroup = s.mixerGroup;
}
```

Eğer ses/sesler belirli bir nesnede oynatılmak isteniyorsa (örn. oyuncu) ses kaynağı bileşeninin ilgili nesnede oluşturulmasını sağlayan fonksiyon.

- **Play(string sound) Fonksiyonu**

```
7 references
public void Play(string sound)
{
    Sound s = Array.Find(sounds, item => item.name == sound);
    if (s == null)
    {
        Debug.LogWarning("Sound: " + sound + " not found!");
        return;
    }

    s.source.volume = s.volume * (1f + UnityEngine.Random.Range(-s.volumeVariance / 2f, s.volumeVariance / 2f));
    s.source.pitch = s.pitch * (1f + UnityEngine.Random.Range(-s.pitchVariance / 2f, s.pitchVariance / 2f));

    s.source.Play();
}
```

Adına göre istenilen sesi çalmamızı sağlayan fonksiyon. Inspector üzerinden yapılan ayarlamalarla sesin hızına ve ya hacmine de rastgelelik bu fonksiyonda uygulanır.

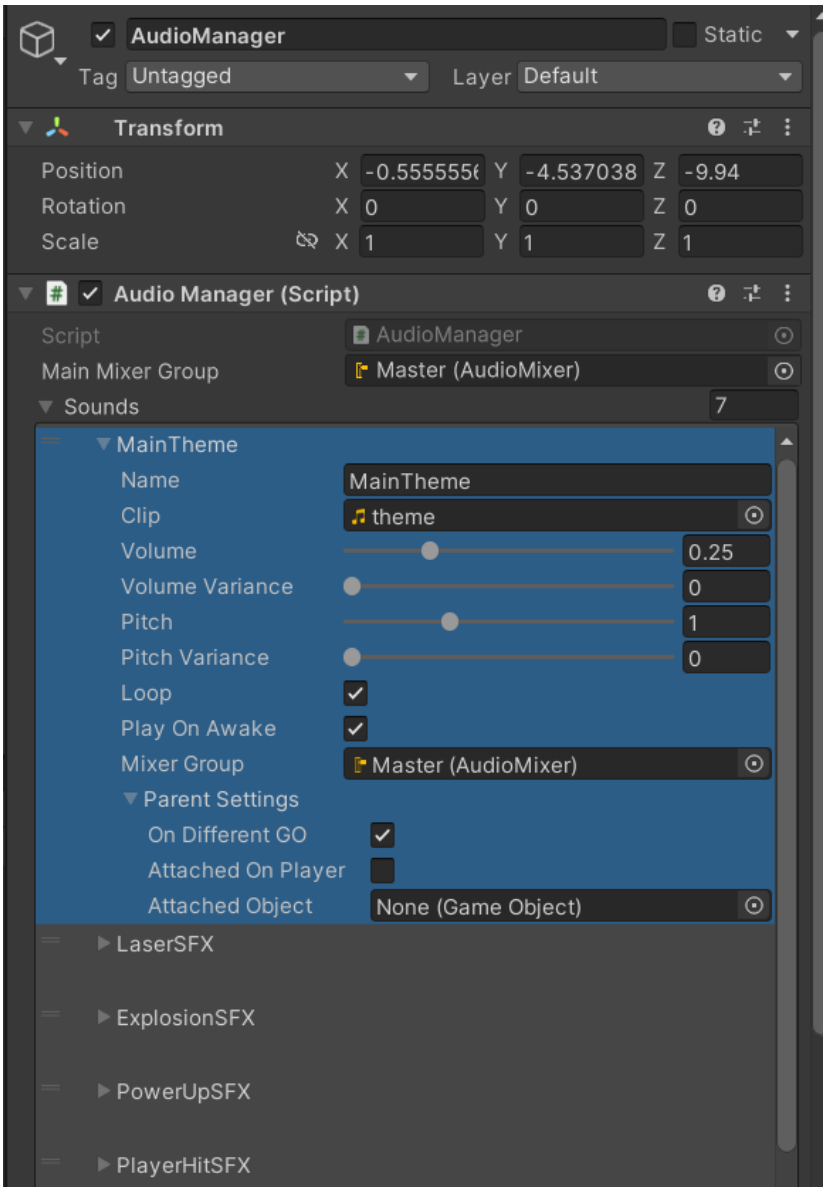
- **PlayAtLocation(string sound, Vector3 pos,float spatialBlendValue,float minRangeValue, float maxRangeValue)**

```
public void PlayAtLocation(string sound, Vector3 pos,float spatialBlendValue,float minRangeValue, float maxRangeValue)
{
    Sound s = Array.Find(sounds, item => item.name == sound);
    if (s == null)
    {
        Debug.LogWarning("Sound: " + sound + " not found!");
        return;
    }

    GameObject tempAudioObject = new GameObject("TempAudio_" + sound);
    tempAudioObject.transform.position = pos;
    tempAudioObject.transform.parent=Referances.Instance.audioContainer.transform;

    AudioSource tempAudioSource = tempAudioObject.AddComponent<AudioSource>();

    tempAudioSource.clip = s.clip;
    tempAudioSource.volume = s.volume * (1f + UnityEngine.Random.Range(-s.volumeVariance / 2f, s.volumeVariance / 2f));
    tempAudioSource.pitch = s.pitch * (1f + UnityEngine.Random.Range(-s.pitchVariance / 2f, s.pitchVariance / 2f));
    tempAudioSource.outputAudioMixerGroup = s.mixerGroup;
    tempAudioSource.spatialBlend = spatialBlendValue;
    tempAudioSource.loop = false;
    tempAudioSource.minDistance = minRangeValue;
    tempAudioSource.maxDistance = maxRangeValue;


    tempAudioSource.Play();

    Destroy(tempAudioObject, s.clip.length / tempAudioSource.pitch);
}
```

Bu fonksiyon da istediğimiz sesi, oyun uzayında istenilen noktada oynatmamıza olanak tanır. Bu fonksiyon patlamalarda kullanılmıştır. Bu sesler 3d ayarlanarak oyuna daha dinamiklik katmıştır.

Oyun İçi Sesler

Yeni bir oyun nesnesi oluşturulup adı “AudioManager” olarak düzenlenmiştir. İçerisine “AudioManager.cs” scripti eklenmiştir.



Yanda görüleceği üzere script bu şekilde gözükmektedir bu inspector görünümü ayrıca özel editor kodları kullanılark yapılmıştır detaylı bilgiler için github reposuna bakabilirsiniz. Sounds sekmesi altında gerekli sesler oluşturulur. Ve ayarları isteğe uygun şekilde yapılır.

- **Arkaplan Müziği**

```
private void Start()
{
    Play("MainTheme");
    SceneManager.sceneLoaded += OnSceneLoaded;
}
```

AudioManager.cs içerisinde ana müzik başlatılır.

- **Lazer**

```
private void HandleShoot()
{

    if(Input.GetKey(KeyCode.Space) && fireCooldown <= 0)
    {
        fireCooldown = timer;
        GameObject laser;
        if (tripleShotIsActive)
        {
            laser = Instantiate(tripleLaserPrefab, transform.position + Vector3.up, Quaternion.identity);
        }
        else
        {
            laser = Instantiate(laserPrefab, transform.position + Vector3.up, Quaternion.identity);
        }
        AudioManager.Instance.Play("LaserSFX");
        laser.transform.parent=Referances.Instance.laserContainer.transform;

        if(laser.TryGetComponent<Rigidbody>(out Rigidbody rb))
        {
            rb.AddForce(Vector3.up * projSpeed,ForceMode.Impulse);
            Destroy(laser, projLifespan);

        }
    }
```

Playermovement.cs içindeki ateş etme fonksiyonunun olduğu yere bu eklenen satır ile ses oynatılır.

- **Patlama Sesi**

```
private void HandleDestruction()
{
    AudioManager.Instance.PlayAtLocation("ExplosionSFX",gameObject.transform.position,.85f,5f,10f);
    if (!isAsteroid)
    {
        animator.SetTrigger("PlayDestroy");
        speed = 0;
        col.enabled = false;
        Destroy(gameObject, 1f);
    }
    else
    {
        speed = speed / 2f;
        col.enabled = false;
        Destroy(gameObject,0.01f);// 0 olduğunda hata oluyor
    }
}
```

Enemy.cs scripti içinde düşmanlar yok edildiğinde çağırılan fonksiyonda yukarıdaki gibi 3 boyutlu olacak şekilde gerekli parametreler verilerek patlama sesi eklenir.

- **Bonus Yakalama Sesi**

```
private void OnTriggerEnter(Collider other)
{
    if (other.CompareTag("Player"))
    {

        switch (type)
        {
            case PowerUpType.TripleShotPowerup:
                pm.StartTriplePowerup();
                break;

            case PowerUpType.SpeedPowerup:
                pm.StartSpeedPowerUp();
                break;

            default:
                Debug.LogWarning("PowerUp type doesnt declared!");
                break;
        }
        AudioManager.Instance.Play("PowerUpSFX");
        Destroy(gameObject);
    }
}
```

PowerUp.cs scripti içerisinde oyuncuyla temasa geçildiğinde çalışan kod içine, ilgili sesi oynatacak kısım eklenmiştir.

Kaynakça

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

https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/UIBasicLayout.html

https://chatgpt.com/

Proje Kodu ve Github Repo

Kod: https://github.com/bathuchan/btu-gameprogramming-BatuhanSengul/tree/main/Reports/10.Hafta

Proje Repo: https://github.com/bathuchan/btu-gameprogramming-BatuhanSengul

Hazırlayan

Batuhan Şengül – 20360859008- bathu.sengul@gmail.com