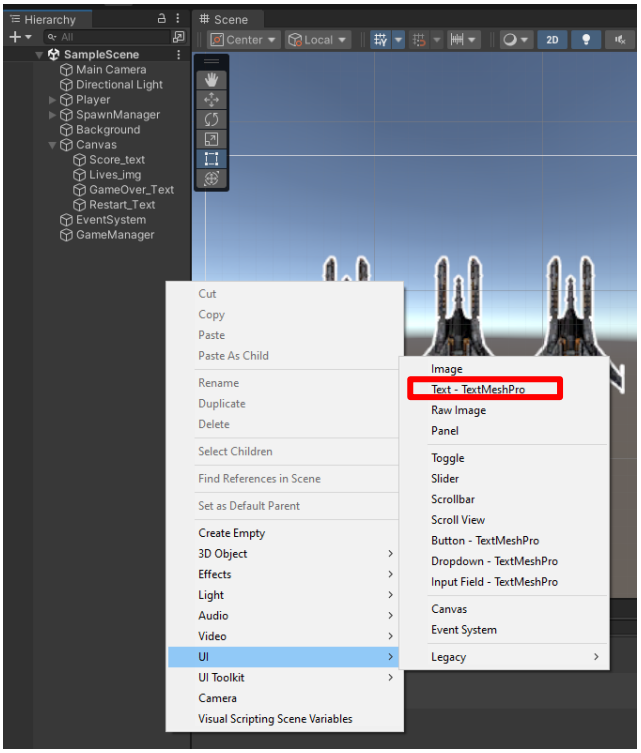
8. ve 9. Hafta Raporu – Aleks Dulda 21360859025

Düşman gemileri yok ettiğinizde puan kazandıracak sistem geliştirin ve anlık puan durumunu ekranda gösterin.

Hierarchy kısmından sağ tıkla “UI” kısmını bulup “Text-TextMeshPro” ekliyoruz. Bunu eklememizle beraber gerekli öğeler hierarchy kısmına eklenmiş oluyor.

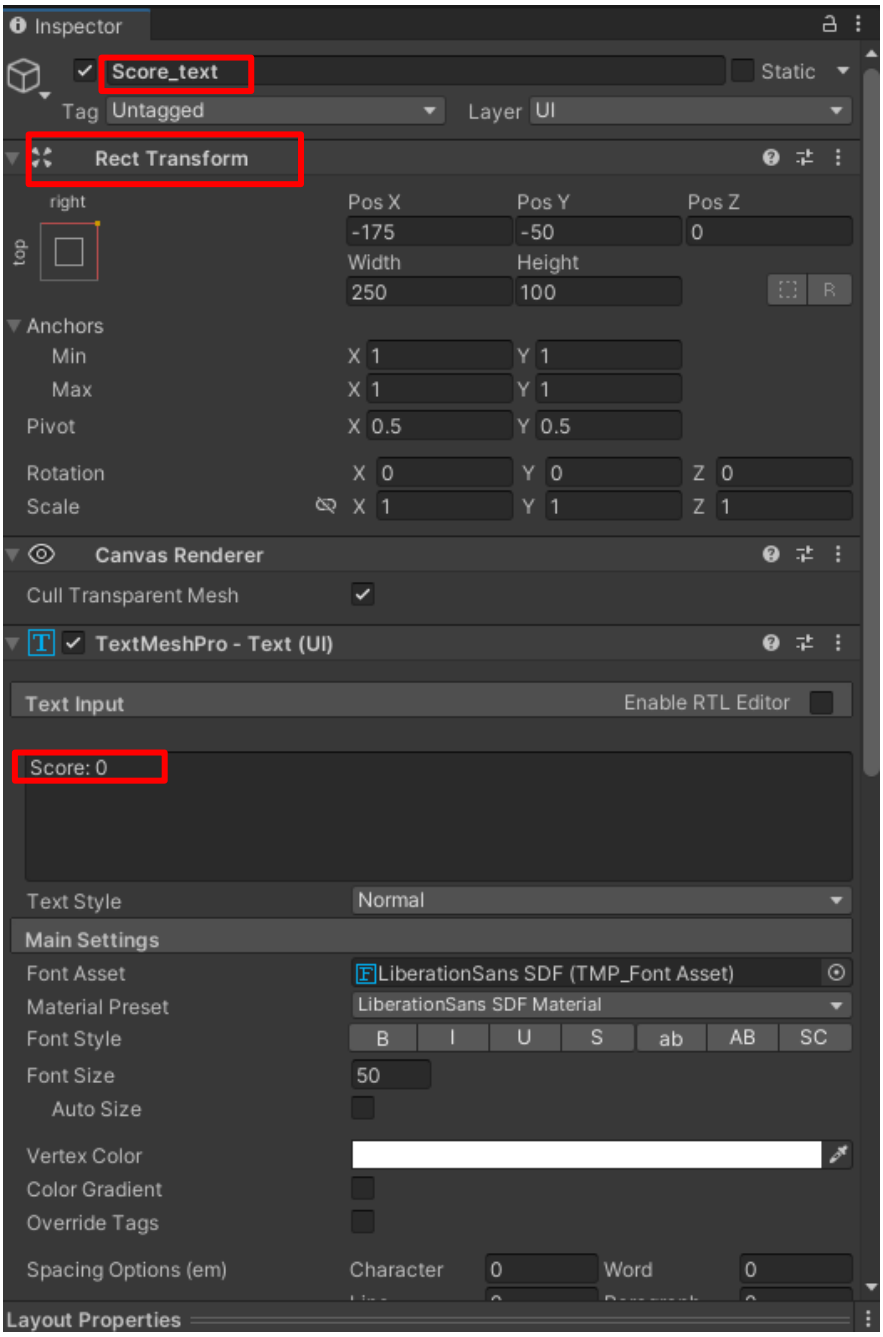


Bahsedilen öğeler Canvas ve EventSystem öğeleridir. TMP öğesini de skorumuzu oluşturmak için kullanacağız.

UIManager_sc script'ini oluşturun ve score değişkeni tanımlayın. Puan arttırma fonksiyonunu oluşturun. Düşman gemisinin laser ile vurulduğunu kontrol edin ve gerekli durumda puan arttırma fonksiyonunu çağırın.

Scane kısmında oluşan büyük çerçevede oluşturduğumuz TMP öğesini'nin konumlandırmasını yapabilirsiniz. Bir diğer seçenek ise "inspector" kısmından oluşturduğunuz öğenin adını boyutunu konumunu içindeki değerleri değiştirebilirsiniz.

Bu aşamada Score_text adlandırması yapılmıştır. Rect Transform kısmından oluşturduğumuz TMP öğesinin nereye sabitleneceğini seçtik. Sağ üst kısma sabitlemeyi seçiyoruz. "Text İnput" kısmından ise içindeki değeri ve gerekli font ayarlamaları bu menüden gerçekleştirebiliyoruz.



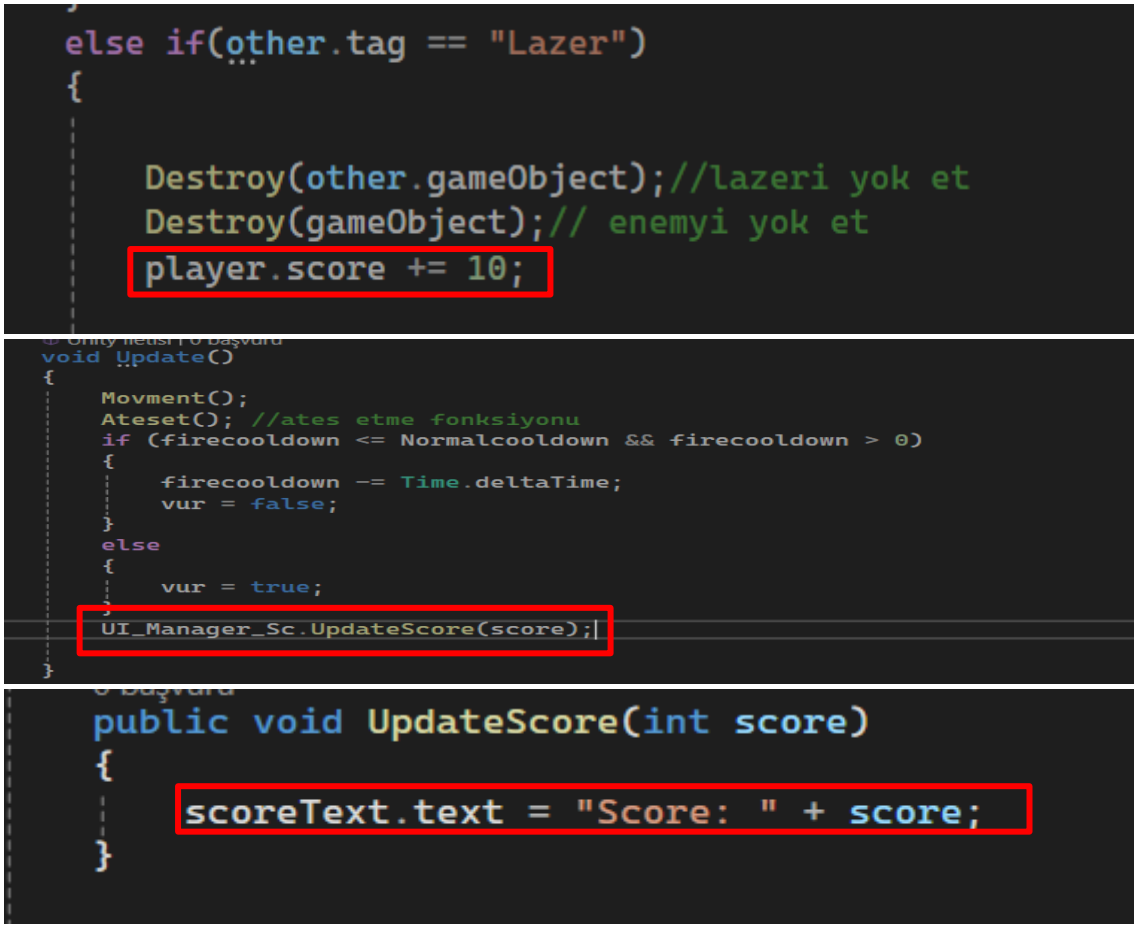
Bir UI_Manager_Sc scripti'ni kullanıcı arayüzünde yapılacak diamik değişimleri kontrol edebilmek için oluşturuyoruz. Bu scriptin içinde Score Lives gibi değişenlerin kontrolleri yapılarak gerekli değişimler sağlanacak.

private TextMeshProUGUI scoreText; şeklinde bir değişken oluşturuyoruz ve bunu [SerializeField] yapıp görünür hale çeviriyoruz. Ardından bir public void UpdateScore(int score) fonksiyonu tanımlayıp scorumuzu güncellemek için içine şunları yazıyoruz soreText.text = "Score: " + score; Bu şekilde fonksiyonumuz çağrıldığı zaman ekranımızdaki score güncellenmiş olacak.

Bu score güncellemesi yapılabilmesi için bu score değerini tutacak değişkeni player_sc dosyamıza koyuyoruz çünkü player ile score bağlantılı varlıklar. Bu player scriptinin update fonksiyonun içinde ise bu scorumuzu güncelleyecek fonksiyonumuzu sürekli çağırıyoruz ki güncel kalabilsin.

Bir başka score ile olan ilişkimiz ise ne zaman arttırılacağı. Bu arttırımı öncelikle enemy_sc scriptindeki enemy ile lazer temasında scorumuzu arttırmaktayız. Onu da şu komutla rahatlıkla yapabiliriz. "player.score += 10;"

Bu sayede gerekli score kullanıcı arayüzümüz oluşmuş oldu.



Oyuna üç canla başlayın ve mevcut canı göstermek için lives sprite'ı kullanın. Can sayısı değiştiğinde kullanılan live sprite'ı güncellenmelidir.

```
[SerializeField]
private Image livesImage;

[SerializeField]
private Sprite[] livesSprites;
```

Ekte de göründüğü üzere bir Image öğesi ve Bir adet liveSprites dizisi oluşturuyoruz. Zaten hali hazırda player scriptimizde bulunan health değişkeni bizim can sayımızı tutmakta. Bu aşamada yapmamız gereken sadece dizimize spriteları eklemek ve bir fonksiyon yazmak olacak.

```
public void UpdateLives(int currentLives)
{
    livesImage.sprite = livesSprites[currentLives];

    if (currentLives == 0)
    {
        GameOverSequance();
    }
}
```

Bu fonksiyonumuz canımızla doğru orantılı olarak kaç canımız var ise o kadar can sayısını gösteren sprite'ı ekranda göstermek için yazılmıştır. Bu şekilde eğer canımız biterse de oyunu bitirme aşaması fonksiyonu çağıralacaktır. Peki bu can fonksiyonu nerde çağrılıyor?

Bu UpdateLives fonksiyonu player scriptinde player hasar aldığı zaman canı azalır ve bu kısımda canımızı güncelleme fonksiyonu çağırılır.

```
1 başvuru
public void Damage()
{
    Health -= 1;
    if (Health <= 0)
    {
        Debug.Log("GAME OVER");
        UI_Manager_Sc.UpdateLives(Health);
        Destroy(gameObject);

    }
    UI_Manager_Sc.UpdateLives(Health);

}
```

Game Over yazısı oluşturun ve oyun bittiğinde ekranda gösterin. Game Over yazısı için flicker etkisi oluşturun. Oyunu yeniden başlatmak için yönerge oluşturun.

Oyuncunun canı bittiği zaman. GameOverSequance(); fonksiyonu çağırılır. Bu fonksiyon oyunun bittiği “GAME OVER“ yazısını ve “Press "R" key to restart” yazısını ekranda aktif eder ve game over yazısı için bir coroutine Flicker çalıştırır. Bu Flicker game over ekrananda yanıp sönemesini sağlar. Bu yanıp sönmeyi yazıyı silio tekrar yazmasıyla aşağıdaki ekteki kod ile yapar.

```
1 başvuru
public void GameOverSequance()
{
    gameoverTMP.gameObject.SetActive(true);
    restartTMP.gameObject.SetActive(true);
    StartCoroutine(GameOverFlicker());
    gameManager.GameOver();
}

1 başvuru
private IEnumerator GameOverFlicker()
{
    while (true)
    {
        gameoverTMP.text = "GAME OVER";
        yield return new WaitForSeconds(0.5f);
        gameoverTMP.text = "";
        yield return new WaitForSeconds(0.5f);
    }
}
```

Dip not: Bu yazıları ekranda oluşturabilmek için score yazısını yaptığımız gibi Hierarchy kısmınadan TMP ekleyerek yapılabilir.

R tuşuna basıldığında oyun yeniden başlasın.

Bu kısmı yapabilmek için gameManager_Sc adında bir script oluşturup oyunun bittiğini analayabileceğimiz bir script oluşturmaktayız.

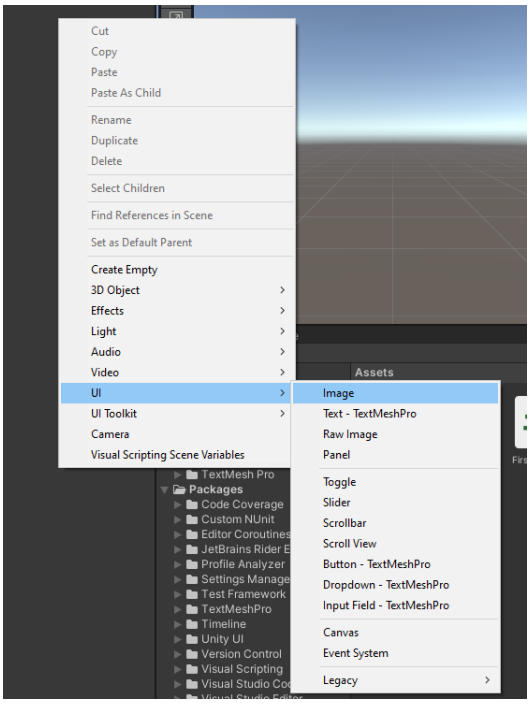
```
public class gameManager_Sc : MonoBehaviour
{
    // Start is called before the first frame update
    bool isGameOver = false;
    Unity İletisi | 0 başvuru
    void Start()
    {

    }
    Unity İletisi | 0 başvuru
    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.R) && isGameOver)
        {
            SceneManager.LoadScene(0);
        }
    }
    1 başvuru
    public void GameOver()
    {
        isGameOver = true;
    }
}
```

Bu kısımda eğer R tuşuna basıldıysa ve oyun bittiyse oyun ekranını tekrardan başlatmasını sağlar. Bu şekilde oyuncunun canı bittiği zaman oyun sahnesini tekrardan başlatmaktadır.

Ana menü için yeni bir sahne oluşturun. Arka planı ayarlayın. Yeni oyun başlatmak için buton ekleyin.

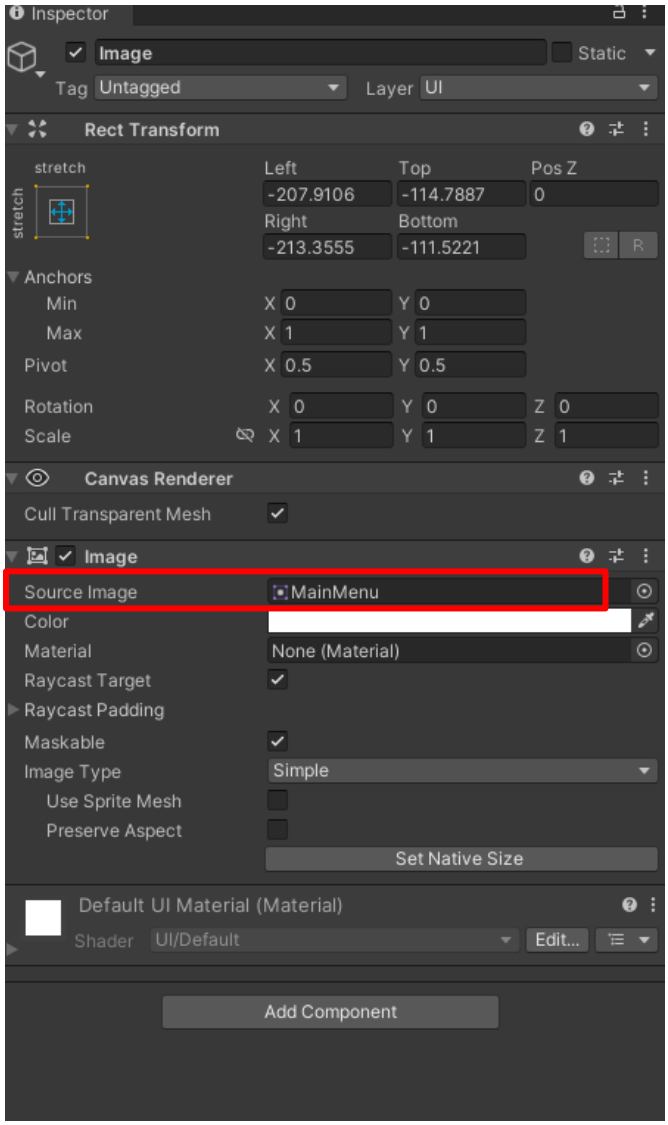
Unity'nin Üst Menü Çubuğunda: File > New Scene seçeneğini seçin. Sahneyi oluşturduktan sonra **Ctrl+S** ile "AnaMenu" gibi bir isimle kaydedin. Sahnemiz oluştu.

Ardından ana ekrana fotoğraf eklemek için



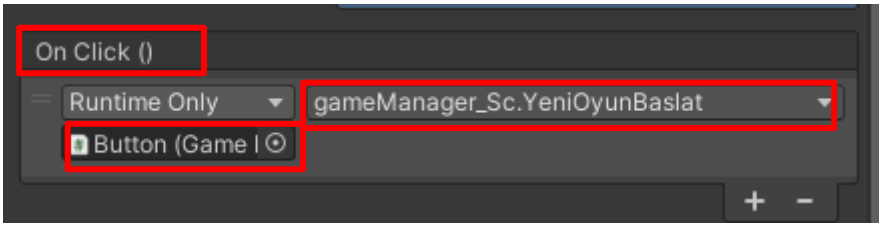
UI kısmından Imageyi seçiyoruz. Bu seçimden sonra Image kısmına dosyalarımızda hazır bulunan scriptimizi koyuyoruz. İstediğimiz şekilde büyüttükten sonra ana ekrana resim ekleme kısmını halletmiş oluyoruz.

Buton eklemek için yine UI kısmından Button – TextMeshPro ekliyoruz. Bu butonumuza tıklandığında fonksiyon ekleyebiliyoruz. Bu fonksiyon diğer sahneye yani oyunumuzun olduğu sahneye gidecek. Bu şekilde bir kişi oyunu oynamak için “New Game” butonunu kullanabilecek.

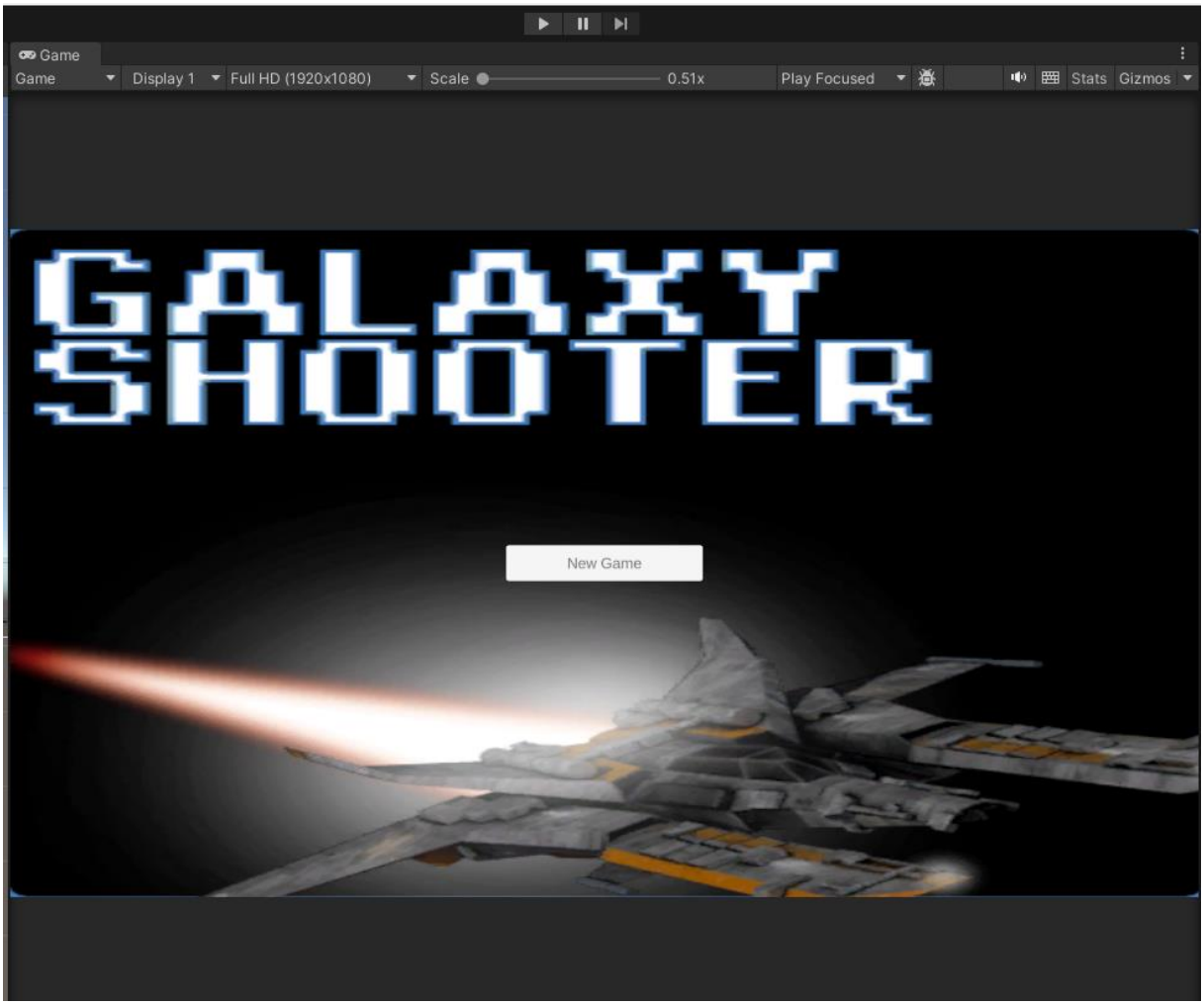


Butona tıklandığında çalışacak komutu gameManager_Sc scriptinde ekte görülen komutu giriyoruz

```
public void YeniOyunBaslat()
{
    SceneManager.LoadScene(0);
}
```



Bu şekilde butona basıldığı zaman bu fonksiyon çağırılacak ve oyun başlamış olacak.



Ana Ekran Görüntüsü

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_8-9