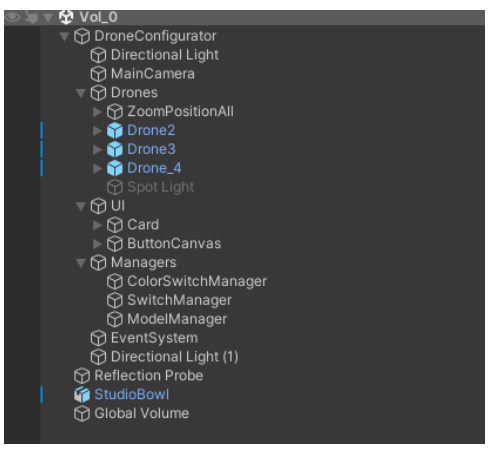
Proje Dokuman:

▼ Vol_0 sahnesi

Vol_0 Sahne yapısı:

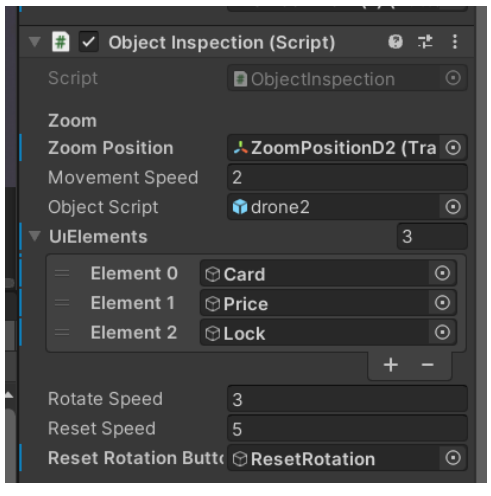
DroneConfigurator/Drones/UI/Managers



Drones:

1. ZoomPositionAll : Drone objelerinin buttona basıldığında kameraya yaklaştığı konumu belirtiyor.

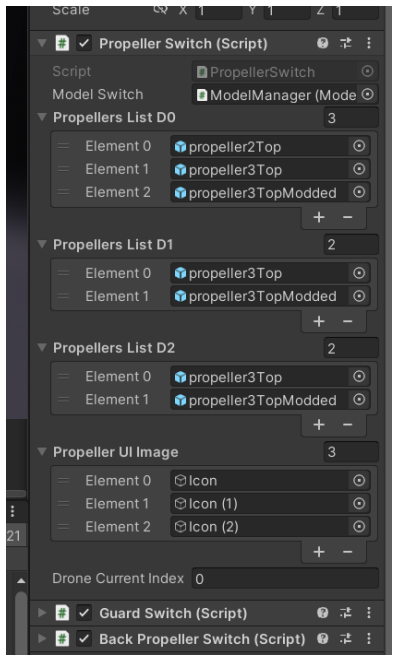
    ▼ Çalıştığı scriptin ss’i



    UIElements listesi: (drone kameraya yaklaştığında ekranda gizlenecek objelerin listesi)

2. Sahnedeki manager yapıları:

    ▼ Örnek: Propeller Switch png ve Kod açıklaması:



Sahnedeki switchlerin kalıtım sayesinde hepsi aynı mimari yapıya sahip. Sürükle bırak mantığı ile çalışıyor.

```
public class PropellerSwitch : ObjectSwitcher
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public abstract class ObjectSwitcher : MonoBehaviour
{
    protected int currentIndex;
    public abstract void ChangeObject(int index);

    public abstract void ToRightChangerFuncButton();


    public abstract void ToLeftChangerFuncButton();

}
```

```
//Hangi dronun aktı olduğu verisi alınıyor
public ModelSwitch modelSwitch;
[SerializeField]
private List<GameObject> propellersListD0, propellersListD1, propellersListD2;
private List<List<GameObject>> propellersLists = new List<List<GameObject>>();
public GameObject[] propellerUIImage;



public int droneCurrentIndex;
Unity Message | 0 references
private void Start()
{
    propellersLists = new List<List<GameObject>>

propellersListD0,
propellersListD1,
propellersListD2

    modelSwitch = GameObject.Find("ModelManager").GetComponent<ModelSwitch>();
    currentIndex = 0;
    ChangeObject(currentIndex);
}
Unity Message | 0 references
private void Update()
{

    droneCurrentIndex = modelSwitch.currentDroneIndex;
}
4 references
public override void ChangeObject(int index)
{
    List<GameObject> propellersList = propellersLists[droneCurrentIndex];

    for (int i = 0; i < propellersList.Count; i++)
    {
        GameObject propeller = propellersList[i];
        propeller.SetActive(i == index);
    }
}

2 references
public void ChangeUIImage(int index)
{

    for (int i = 0; i < propellerUIImage.Length; i++)
    {
        GameObject propeller = propellerUIImage[i];
        propeller.SetActive(i == index);
    }
}



1 reference
public override void ToRightChangerFuncButton()
{
    List<GameObject> propellersList = propellersLists[droneCurrentIndex];
    currentIndex++;
    if (currentIndex >= propellersList.Count)
    {
        currentIndex = 0;
    }
    ChangeObject(currentIndex);
    ChangeUIImage(currentIndex);
}

1 reference
public override void ToLeftChangerFuncButton()
{
    List<GameObject> propellersList = propellersLists[droneCurrentIndex];
    currentIndex--;
    if (currentIndex < 0)
    {
        currentIndex = propellersList.Count-1;
    }
    ChangeObject(currentIndex);
    ChangeUIImage(currentIndex);
}
```

1. Propelerları 3 listeye böldüm. Bu bölme işlemi her dronun propellerlarının farklı sayıda ve yapıda olması.
2.Bunları bir üst liste içerisinde tuttum.
3. Bu sayede 1. dronun propelerlarının olduğu liste 0. indexte olacak. Aktif dronun indexi 0 geldiğinde direct olarak propellerına ulaşabiliyoruz.

Yeni bir drone eklemesi yapmak için

```
Unity Script (2 asset references) | 16 references
public class ModelSwitch : MonoBehaviour
{
    public List<GameObject> drones = new List<GameObject>(); // Farklı drone'ları tutan liste
    public int currentDroneIndex = 0; // Şu anki drone'un indeksi
```

drones listesine dronu sürüklemek.

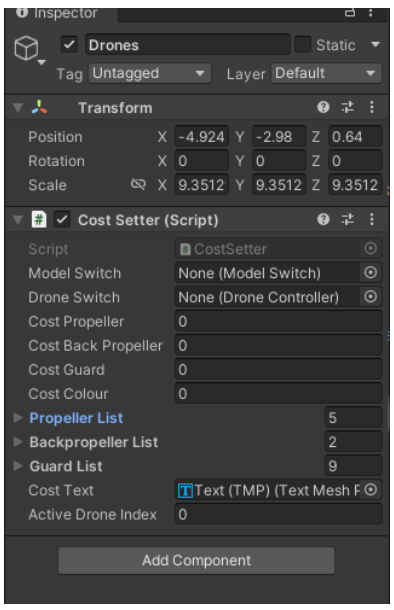
Daha sonra diğer managerlara gidip parçalarının listesini altta görünen yere eklemek yeterlidir.

```
        propellersLists = new List<List<GameObject>>
    {
        propellersListD0,
        propellersListD1,
        propellersListD2
    };
```

3. Para hesabı:

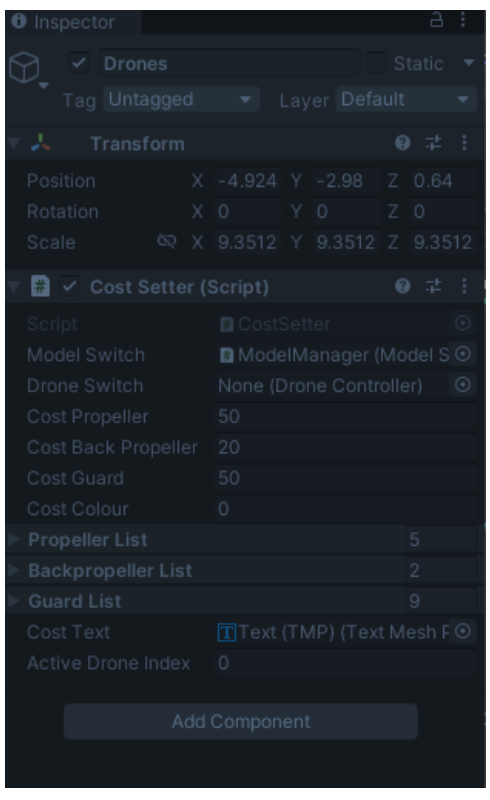
▼ Çalışması ve scriptleri

Bu objenin üzerindeki scripten yapılıyor.



Verilerin kontrol edildiği ve yazıldığı ana script.

Parasını hesaplayacağınız objenin üzerine attığınız scrip.



Çalışırken ki hali.

▼ Vol_1 sahnesi

Vol_1 Sahne yapısı:

1. Önceki sahne ile aynı yapıya sahip.

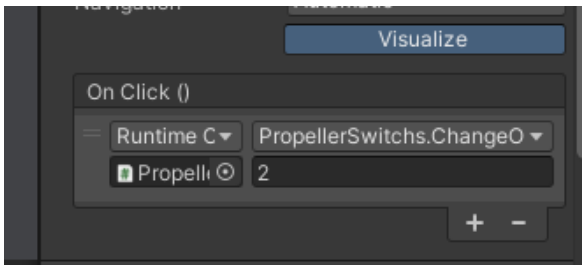
2. Sahnedeki manager yapsı:

▼ Manager Genel çalışma mantığı

```
public class PropellerSwitchs : MonoBehaviour
{
    // Başka bir DroneController bileşenine referans sağlar.
    public DroneController modelSwitch;
    // Farklı pervane gruplarını içeren listeler.
    [SerializeField] private List<GameObject> propellersListD0, propellersListD1, propellersListD2;
    // Tüm pervane gruplarını içeren ana liste.
    private List<List<GameObject>> propellersLists = new List<List<GameObject>>();
    // Mevcut pervane grubunun indeksini belirtir.
    public int currentIndex;
    // Arka plan görüntülerini içeren dizi.
    public Image[] backGround;
    // Arka plan rengini belirlemek için kullanılır.
    public Color color;
    public int droneCurrentIndex;

    Unity Message | 0 references
    private void Start()
    {
        propellersLists = new List<List<GameObject>>
{
    propellersListD0,
    propellersListD1,
    propellersListD2
};
        modelSwitch = GameObject.Find("DroneManager").GetComponent<DroneController>();
        ChangeObject(0);
    }
    Unity Message | 0 references
    private void Update()
    {
        droneCurrentIndex = modelSwitch.currentDroneIndex;
    }
    1 reference
    public void ChangeObject(int index)
    {

        #region BackGroundColorChance
        // Önceki dronun rengini eski haline getir
        backGround[currentIndex].color = Color.white;
        // Yeni dronun rengini ayarla
        backGround[index].color = color;

        currentIndex = index;
        // Önceki dronun rengini ayarla
        if (currentIndex > 0)
        {
            backGround[currentIndex - 1].color = Color.white;
        }

        // Sonraki dronun rengini ayarla
        if (currentIndex < backGround.Length - 1)
        {
            backGround[currentIndex + 1].color = Color.white;
        }
        #endregion


        //Propeller değiştirmeye yarayan kısım
        List<GameObject> propellersList = propellersLists[droneCurrentIndex];

        for (int i = 0; i < propellersList.Count; i++)
        {
            GameObject propeller = propellersList[i];
            propeller.SetActive(i == index);
        }
    }
```

Yapı olarak aynı fakat çalışma şekli farklı.

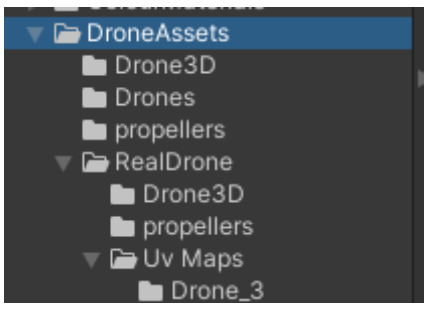
1. ChangeObject metodunda işlemi yapıyorum ve aldığım değer sonucu listede eşleşen objeyi görünür yapıyorum.



gelen veri değeri listedeki 2. objeyi istiyorum demek.

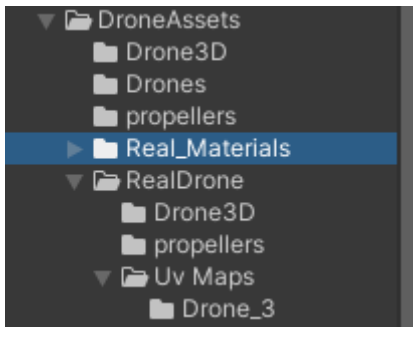
3. Sahnedeki UI animasyonu için ve arka plan renk değişimleri için DotWeen kullandım.

▼ Proje Karışık dosya yapısı!!



Real drone sahnede olan dronlar.

Fakat düzensiliklten bazı dron parçaları diğer klasörlerde olabiliyor. Hepsini bu klasör altında topladım.

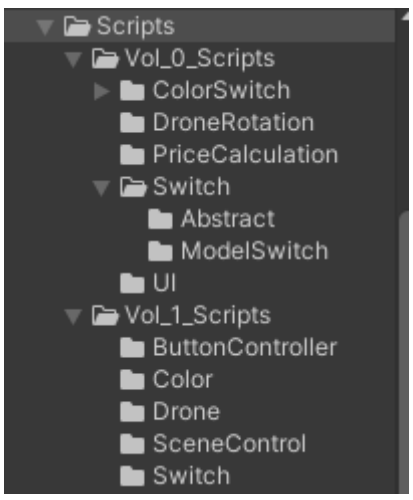


Sahnede asıl kullanılan materialler. Drone materialleri ve shaderlar.

Bu kısım karışık olan kısım.

Buradaki matariallerin çok azı sahnelerde var bunu tespit etmek zor olduğundan silmedim. Temizlemek her seferinde sorun çıkardı.

▼ Scripts Dosyası Yapısı.



Her Sahnenin kodu ayrılmış VE düzenlenmiş.

Link: Notion

https://vagabond-yak-b2c.notion.site/Proje-Dokuman-225b884a44de481da6bdc6e1b8b30f9f?pvs=4

L