





**PAMUKKALE ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ**

**İstanbuldan Kaçış**

**LİSANS TEZİ**

**Caner Savak  
(16253061)**

**Bilgisayar Mühendisliği**

**Tez Danışmanı: Öğr.Gör. Şevket Umut Çakır**

**Haziran 2021**



Pamukkale Üniversitesi Bilgisayar Mühendisliği Bölümü, 16253061 numaralı Lisans Öğrencisi Caner Savak, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı “İstanbuldan Kaçış” başlıklı tezini aşağıdaki imzaları olan jüri önünde başarı ile sunmuştur.

**Tez Danışmanı :**      **Öğr.Gör. Şevket Umut Çakır**      .....

**Jüri Üyeleri :**              **Dr. Öğrt. Üyesi Gökçen Yılmaz**      .....  
Pamukkale Üniversitesi

**Arş. Gör. Mine Yoldaş**      .....  
Pamukkale Üniversitesi

**Teslim Tarihi :**      **4 Haziran 2021**

**Savunma Tarihi :**      **4 Haziran 2021**



## **Önsöz**

Tez çalışmama karar verdikten sonra, sunduğum başta tez danışmanım Öğr.Gör.Şevket Umut ÇAKIR olmak üzere diğer jüri üyeleri öğretmenlerimin çalışmamın geliştirilmesi konusundaki fikirleri, yardımları ve önerileri için teşekkür ederim.

Haziran 2021

Caner Savak





# İçindekiler

## Sayfa

ÖNSÖZ .....	v
İÇİNDEKİLER .....	vii
ÇİZELGE LİSTESİ.....	ix
ŞEKİL LİSTESİ.....	xi
ÖZET .....	xiii
SUMMARY .....	xv
<b>1. GİRİŞ .....</b>	<b>1</b>
1.1 Tezin Amacı.....	1
1.1.1 Tezin ikincil amaçları .....	1
1.2 Literatür Araştırması .....	1
1.2.1 Subway Surfers.....	1
1.2.2 Temple Run .....	3
<b>2. UYGULAMANIN GÖRSEL TASARIMI .....</b>	<b>5</b>
2.1 Yan Ortam Görsel Tasarımları.....	5
2.1.1 Cami 3D Objesi .....	5
2.1.2 Ağaç 3D Objesi .....	6
2.1.3 Mahalle Evi .....	6
2.1.4 Lüks Semt Evi .....	7
2.1.5 Galata Kulesi .....	7
2.1.6 Gemi 3D Objesi ve Marmara Denizi.....	8
2.1.7 Kız Kulesi.....	8
2.1.8 15 Temmuz Şehitler Köprüsü .....	9
2.2 Yol Tasarımı.....	9
2.3 Oyun Karakter Tasarımı .....	10
2.3.1 Animatör Oluşturulması Ve Değişkenler.....	11
2.3.2 Karakter Vücut Yapısı .....	12
2.3.3 Animator Player Ataması .....	13
<b>3. UYGULAMANIN ENGEL VE ÖDÜL SİSTEMİ .....</b>	<b>15</b>
3.1 Ödül Sistemi .....	15
3.1.1 Coin Sistemi .....	15
3.1.2 Mıknatıs Sistemi.....	16
3.1.3 Temas Küpü.....	17
3.1.4 Puan Sistemi .....	17
3.2 Engel Sistemi.....	18
3.2.1 Hareketsiz Engel.....	18
3.2.2 Hareketli Engel .....	19

<b>4. OYUN KODLAMASI.....</b>	<b>21</b>
4.1 Düşme Engeli .....	21
4.2 Karakter Hareket Sistemi .....	21
4.2.1 Zıplama Sistemi.....	22
4.3 Karakter Kodlaması.....	23
4.3.1 Karakter Değişken Ve Game Object Atamaları.....	24
4.3.2 Sonsuz Yol Döngüsü ve Çarpışma Sistemi .....	24
4.3.3 Zıplama Sesi Ve Mıknatıs Kontrol .....	25
4.3.4 Puan Artımı .....	26
4.4 Engel Ve Ödül Klonlama.....	26
4.4.1 Klonlama Kodu.....	26
4.4.2 Klonlama Oluşturma .....	27
4.4.3 Tekli Klon Ve Konumu Belirleme .....	27
4.4.4 Çoklu Klon Adedi Ve Konumu Belirleme .....	28
4.4.5 Klon Atamaları .....	29
4.5 Kamera Takip Sistemi .....	29
<b>5. MENÜ VE BİTİŞ EKRANI.....</b>	<b>31</b>
5.1 Menü Oluşturma Kodu .....	31
5.1.1 Menü Görsel Görünüm.....	32
5.1.2 Menü Button Atamaları .....	32
5.2 Bitiş Panel Kod.....	33
5.2.1 Bitiş Paneli Görünüm .....	33
<b>6. KULLANILAN TEKNOLOJİLER .....</b>	<b>35</b>
6.1 Visual Studio .....	35
6.2 Unity .....	36
6.3 Unity Asset Store .....	37
6.4 Adobe Animate.....	37
6.5 Adobe Fuse.....	38
<b>7. UYGULAMADA YAPILABİLECEK GELİŞTİRMELER.....</b>	<b>39</b>
<b>KAYNAKLAR.....</b>	<b>41</b>
<b>ÖZGEÇMİŞ .....</b>	<b>43</b>





# Şekil Listesi

	<u>Sayfa</u>
Şekil 1.1 : Subway Surfers.....	2
Şekil 1.2 : Temple Run.....	3
Şekil 2.1 : İstanbul Eyüp Sultan CAMİ .....	5
Şekil 2.2 : Ağaç Objesi .....	6
Şekil 2.3 : İstanbul Mahalle Arası Ev .....	6
Şekil 2.4 : İstanbul Lüks Semt Evi.....	7
Şekil 2.5 : Galata Kulesi .....	7
Şekil 2.6 : Yolcu Gemisi ve Marmara Denizi .....	8
Şekil 2.7 : Kız Kulesi .....	8
Şekil 2.8 : Boğaz Köprüsü .....	9
Şekil 2.9 : Koşulan Cadde.....	9
Şekil 2.10 : Oyun Karakteri .....	10
Şekil 2.11 : Animator.....	11
Şekil 2.12 : Animator Değişkenleri .....	11
Şekil 2.13 : Karakter Rigidbody .....	12
Şekil 2.14 : Karakter Collider .....	12
Şekil 2.15 : Animator Player Bağlaması.....	13
Şekil 3.1 : TL Coin.....	15
Şekil 3.2 : Mıknatıs .....	16
Şekil 3.3 : Coin Mıknatıs İlişkisi Kod.....	16
Şekil 3.4 : Karakterimizin Temas Küpü.....	17
Şekil 3.5 : Oyun İlerleme Durum Ekranı.....	17
Şekil 3.6 : Yol Yapım Engelleri .....	18
Şekil 3.7 : Basit Beton Bariyer .....	18
Şekil 3.8 : Çöp Kutusu .....	19
Şekil 3.9 : Araba .....	19
Şekil 3.10 : İtfaiye Kamyonu .....	20
Şekil 3.11 : Hareketli Engel Kod .....	20
Şekil 4.1 : Düşme Engeli .....	21
Şekil 4.2 : Karakter Hareket Kodu.....	21
Şekil 4.3 : Karakter Hareket Kodu.....	22
Şekil 4.4 : Karakter Hareket Kodu.....	22
Şekil 4.5 : Karakter Kodu .....	23
Şekil 4.6 : Player Atamaları .....	24
Şekil 4.7 : Sonsuz Döngü Ve Çarpışmalar Kodu .....	25
Şekil 4.8 : Mıknatıs Kontrol Ve Ses Kontrol Kodu.....	25

<b>Şekil 4.9</b>	: Mıknatıs Kontrol Ve Ses Kontrol Kodu.....	26
<b>Şekil 4.10</b>	: Klonlama Kod .....	26
<b>Şekil 4.11</b>	: Klon Oluşturma .....	27
<b>Şekil 4.12</b>	: Klon Oluşturma Ve Konum .....	28
<b>Şekil 4.13</b>	: Klon Oluşturma Ve Konum .....	28
<b>Şekil 4.14</b>	: Klon Atamaları .....	29
<b>Şekil 4.15</b>	: Kamera Takip Kodu .....	29
<b>Şekil 5.1</b>	: Sahneler .....	31
<b>Şekil 5.2</b>	: Menü Kod .....	31
<b>Şekil 5.3</b>	: Menü .....	32
<b>Şekil 5.4</b>	: Menü Ayarlar Kısım .....	32
<b>Şekil 5.5</b>	: Buton Atama.....	32
<b>Şekil 5.6</b>	: Buton Atama.....	32
<b>Şekil 5.7</b>	: Bitiş Paneli Kodu.....	33
<b>Şekil 5.8</b>	: Bitiş Paneli Görünümü .....	33
<b>Şekil 6.1</b>	: Visual Studio.....	35
<b>Şekil 6.2</b>	: Unity .....	36
<b>Şekil 6.3</b>	: Unity Asset Story.....	37
<b>Şekil 6.4</b>	: Adobe Animate.....	38
<b>Şekil 6.5</b>	: Adobe Fuse .....	38

## İstanbuldan Kaçış

### Özet

Mobil teknolojilerindeki hızlı gelişmeler sonucunda geleneksel bilgisayar uygulamaları yerini mobil uygulamalara bırakmaya başlamaktadır. Bu uygulamalar mobil platformlar sayesinde her zaman erişilip kullanılabilir niteliğe bürünmektedir. Özellikle bilgisayarları dijital oyun aracı olarak kullanan bireyler için mobil cihazlar günümüzde taşınabilir ve her ortamda oynanabilir olmasından dolayı daha çok tercih edilmektedir. Bu bağlamda araştırmanın temel amacı kullanımı günden güne artan mobil oyun platformunda ülkemizden bir şehir olan İstanbul'dan kesitlerle belli alanlarda ilerleyen bir aksiyon oyunu yaparak insanların keyifli ve rahat zaman geçirmesine olanak sağlamak. Oyunda seçtiğimiz karakter ile havaalanına ulaşmaya çalışıyoruz. Yolda ve denizde olmak üzere iki yerden ilerlenilecek oyunun arka planında İstanbul'dan tarihi yerler olacak. Oyunumuzda yüksek skor ve oyun içi para olacak ve bu paralarla çeşitli karakterlere ulaşmak mümkün olacak.

**Anahtar Kelimeler:** Mobil oyun, android, ios, apk, veri tabanı, istanbul, kaçış oyunu, yarışma, engel atlama, parkur, eğlence , offline oyun,





## **İstanbuldan Kaçış**

### **SUMMARY**

As a result of the rapid developments in mobile technologies, traditional computer applications are starting to leave their place to mobile applications. These applications are always accessible and usable thanks to mobile platforms. Especially for individuals who use computers as a digital game tool, mobile devices are more preferred today because they are portable and can be played in any environment. In this context, the main purpose of the research is to enable people to have a pleasant and comfortable time by making an action game that progresses in certain areas with sections from Istanbul, a city in our country, on the mobile game platform, the use of which is increasing day by day. We are trying to reach the airport with the character we chose in the game. There will be historical places from Istanbul in the background of the game, which will be progressed from two places, on the road and at the sea. There will be high score and in-game money in our game and it will be possible to reach various characters with these money.

**Keywords:** Mobile game, ios, apk, istanbul, escape game, jumping, database, parkour,offline oyun, entertainment, android, competition



## **1. GİRİŞ**

Bilgisayarlar üzerinde yapılan işlemlerin birçoğu akıllı telefon, tablet gibi mobil cihazlar tarafından gerçekleştirilmekte olup bilgisayara duyulan ihtiyaç her geçen gün azalmaktadır. Özellikle günlük internet kullanım süresinin günlük bilgisayar kullanım süresinden daha fazla olması sonucu mobil cihazlara istek daha da arttı. Özellikle gençler başta olmak üzere mobil cihazları dijital oyun aracı olarak kullanan kişilerin sayısı bir hayli fazladır. Dijital oyunların eğlenceli, kişisel tercihlerin öne çıkması hedef kitle için tercih nedenleri arasındadır. İnsanlar kişisel tercihlerinin yanında oyundan kendinden bir parça da bulmak istiyor. Bu çalışmamda ülkemizin şehirlerinden İstanbulu avuçları içine getirerek onlara zevkli, eğlenceli saatler geçirtmek istiyorum.

### **1.1 Tezin Amacı**

İnsanların boş zamanlarını veya eğlenmek istediği zamanlarda onlara istediklerini vererek eğlenceli hoş zaman geçirtmek ve oyun içerisindeki gerçek yapılardan alınarak göremediği veya gördüğü yerleri zihninde canlandırmak.

#### **1.1.1 Tezin ikincil amaçları**

Oyunumuza ilk giriş ekranında her hangi bir profil seçimi yapılarak giriş yapılır. Daha sonra oyun içinden seçeceğimiz karakterimizle oyuna başlayıp karşımıza çıkan engellere çarpmadan aynı zamanda bazı kısımlarda karadan bazı kısımlarda denizde gidilirken yol üzerindeki oyun altınlarını da toplayarak oyunumuzu bitirmeye çalışıyoruz. Topladığımız altınlar oyun içinde farklı karakterleri açmamıza olanak sağlıyor.

### **1.2 Literatür Araştırması**

#### **1.2.1 Subway Surfers**



Şekil 1.1 : Subway Surfers

Sonsuz koşu türündeki oyunlar arasında yer alan Subway Surfers, bir gencin Dünya'nın farklı şehirlerinde grafiti yaparken bekçiye yakalanmasını ve metro raylarında koşarak ya da kaykay kullanarak bekçi ve köpeğinden kaçarken en fazla ödülün toplanmaya çalışılmasını konu edinmektedir. Oyundaki bu farklı ve popüler şehirler, zaman zaman yapılan güncelleştirmelerle değiştirilmekte ve oyunda şehir seçme hakkı bulunmamaktadır. Oyunda karşılaşılan trenler ve çeşitli engeller, üstünden atlayarak, altından yuvarlanarak ya da yanındaki diğer yolları kullanarak aşmaya çalışılmaktadır. Oyunda kazanılan sanal altınlar harcanarak ve farklı görevler

tamamlanarak kaykay modellerinin ve farklı karakterlerin kilidi açılabilmekte, özel güçler geliştirilebilmektedir.

### 1.2.2 Temple Run



Şekil 1.2 : Temple Run

Temple Run isimli oyun Android telefonlar için ücretsiz bir oyundur. Oyunda bir mağaradan çıkan kahramanımızı goril kovalamaya başlıyor. Sizde koşucu kahramanınızı yönlendiriyorsunuz. Karşınıza çıkan engellerden ise zıplayarak, eğilerek ve sağa sola giderek kurtuluyorsunuz. Aksi takdirde kahramanımız ya sulara çakılıyor ya da sağa sola tosluyor. Oyunu eğlence ve rekabet için oynayabilirsiniz. Oyundaki tek amacınız koşmak ve olabildiğince fazla puan toplamaktır. Böylece milyonlarca oyuncu arasında en iyi skorları elde edebilirsiniz.



## 2. UYGULAMANIN GÖRSEL TASARIMI

### 2.1 Yan Ortam Görsel Tasarımları

Uygulamamızın temel amacı olan İstanbul şehrinin bilinen ve tanınan yapılarının 3 boyutlu modellerini koyma işlemlerimiz bu şekildedir. Oyunumuzu oynarken gerçekten İstanbul sokaklarında koşuyormuş hissiyatını ve keyfini oyunumuzu oynayan kişilere aktarmak istedim. Bu amaç doğrultusunda İstanbulun tanınmış yapılarını 3 boyutlu bir şekilde özenle seçtim. Bu doğrultuda oyuncularımızın oyun zevkini yapmış olduğum tasarımla görsel şölene tabi tutuyorum. Diğer oyun uygulamalarından beni ayırtan ve öne geçiren bir tasarım yapmaya çalıştım.

#### 2.1.1 Cami 3D Objesi



Şekil 2.1 : İstanbul Eyüp Sultan CAMİ

Eyüp Sultan Camii, İstanbul'da Eyüpsultan semtinde Haliç kıyısında bulunan cami, olmasının ötesinde kutsal bir ziyaret yeridir.



### 2.1.2 Ağaç 3D Objesi



Şekil 2.2 : Ağaç Objesi

### 2.1.3 Mahalle Evi



Şekil 2.3 : İstanbul Mahalle Arası Ev



#### 2.1.4 Lüks Semt Evi



Şekil 2.4 : İstanbul Lüks Semt Evi

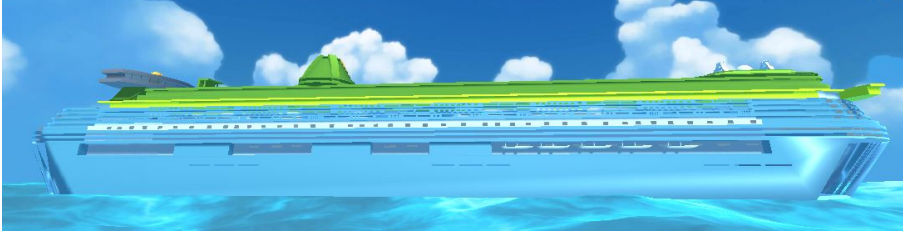
#### 2.1.5 Galata Kulesi



Şekil 2.5 : Galata Kulesi

Dünyanın en eski kuleleri arasında sayılan ve İstanbul'un sembollerinden biri olan Galata Kulesi, 2013 yılında UNESCO Dünya Mirası Geçici Listesi'ne dahil edilmiştir. İstanbul'un silüetini oluşturan en önemli yapılardan biri olan Galata Kulesi, uzun dönem yangın gözetleme kulesi olarak kullanıldı.

### 2.1.6 Gemi 3D Objesi ve Marmara Denizi



Şekil 2.6 : Yolcu Gemisi ve Marmara Denizi

### 2.1.7 Kız Kulesi



Şekil 2.7 : Kız Kulesi

Boğaz manzarasının vazgeçilmez yerlerinden biri de kuşkusuz Kız Kulesi'dir. Salacak açıklarındaki küçük bir adanın üzerine inşa edilmiş olan kule, pek çok efsaneye konu olmaktadır. Bunlardan biri ve en bilineni, kuleye adını da vermiş olan Leandros efsanesi'dir. Aralarındaki denize meydan okuyan aşıklar Leondros ve Hero'nun trajedik hikayesinin yansımasıdır.

### 2.1.8 15 Temmuz Şehitler Köprüsü



Şekil 2.8 : Boğaz Köprüsü

15 Temmuz Şehitler Köprüsü, eski adıyla Boğaziçi Köprüsü ya da boğaza inşa edilen ilk köprü olmasına atıfla Birinci Köprü; Karadeniz ile Marmara Denizini birbirine bağlayan İstanbul Boğazı üzerinde yer alan üç asma köprüden biridir.

### 2.2 Yol Tasarımı



Şekil 2.9 : Koşulan Cadde

Karakterimizin oyun boyunca koşacağı ortamı cadde havası vermesi ve engellerin oluşabileceği yer olarak anlamlı olabilmesi açısından asfalt yol tercihi ettim.

### 2.3 Oyun Karakter Tasarımı

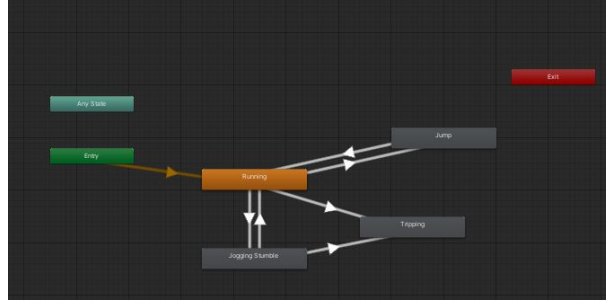
Oyunda oluşturacağım karakteri hem ilgi çekici hem de keyiflendirici bir tip olarak belirledim. Aynı zamanda karakterimiz İstanbul içinde koşacağı için o şehrin insan profiline uygun olmalı. Oyunu oynayan genel insan profilini tahmin ederekten oyunu oynayan insanların hem karakteri komik ve ilgi çekebilecek şekilde ayarladım. Oyun



Şekil 2.10 : Oyun Karakteri

karakterini adobenin mixamo programından hazırlanarak yapılmıştır. Daha sonra oyun içinde gerekli olacak animasyonlar yine mixamodan atamaları yapılarak tanıtılmıştır. Koşma veya zıplama animasyonu gibi. Unity programında bu animasyonlarımıza yol şeması çizilerek ve çalışma prensipleri belirlenerek gerekli atamaları yapılarak animatör oluşturulmuştur.

### 2.3.1 Animatör Oluşturulması Ve Değişkenler



Şekil 2.11 : Animator

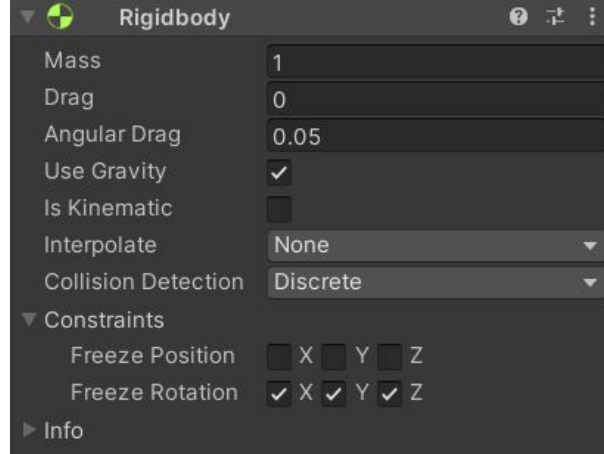
Hazırlanan bu animatöre mantıklı çalışma olanağı sağlaması için bazen sürekli animasyonu gerçekleştirmesi bazı durumlarda bir kere gerçekleştirip bitirmesi gerekmektedir.Örneğin karakterimizin oyun boyunca koşması gerekmektedir ama zıplarken koşma animasyonunun kesilmesi aynı zamanda zıplarken zıplama animasyonunun da bir kere çalışması gerekmektedir. Daha sonra yere düştüğünde tekrar koşma animasyonuna devam etmesi gerekir. Bu olayları kontrol etmek amaçlı animatörümüze belirli trigger ve bool değişkenleri oluşturulmuştur.

hit	<input type="radio"/>
killed	<input type="radio"/>
Jump	<input type="checkbox"/>
killed1	<input checked="" type="checkbox"/>

Şekil 2.12 : Animator Değişkenleri

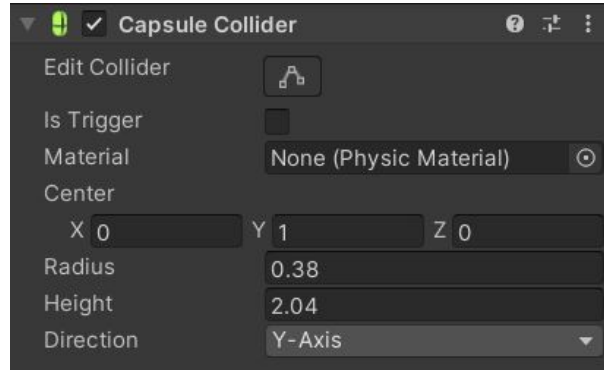
### 2.3.2 Karakter Vücut Yapısı

Karakterimizin oyun ortamı bir cadde üzerinde koştuğu için gerçeklik içeriyor bundan dolayı yer çekimi gibi var olan etkilerden etkilenmesi gerekmektedir. Bu olayı sağlamak için karakterimize rigidbody özelliği ekleyerek gravityyi enabled yapıp gerekli rotasyonlarda durması gerektiğini belirtiyoruz. Karakterimizin ortamdaki



Şekil 2.13 : Karakter Rigidbody

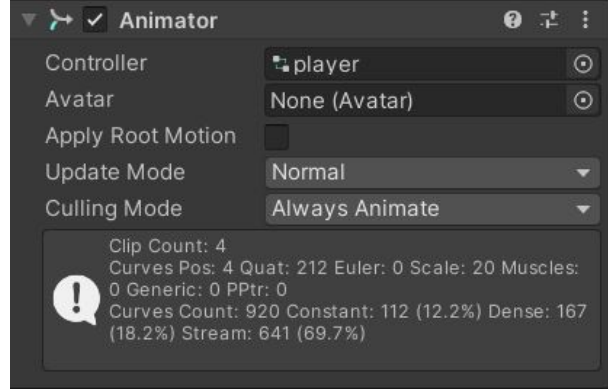
nesnelerin bazılarının (mıknatıs,coin) içinden geçerek toplaması gerektiği gibi engellere çarptığında ise yanarak kaybetmesi gerekmektedir. Bu çarpışma olaylarının belirlenmesi için karakterimize collider eklenerek gerekli vücut bölgelerine göre atamalar yapılmıştır.



Şekil 2.14 : Karakter Collider

### 2.3.3 Animator Player Ataması

Oluşturulan animatörümüzü karakterimize bağlayarak gerekli bağlantıyı kuruyoruz.



Şekil 2.15 : Animator Player Bağlaması





### 3. UYGULAMANIN ENGEL VE ÖDÜL SİSTEMİ

Uygulamamızın sürekliliği ve amacı olması gerektiğinden belirli puan ve yol üzerinden alınacak ödüller tasarlanmıştır. Bu alınan puanlar oynayan kişilerin rekabetini arttırarak kullanıcıları bir yarış moduna sokarak oyunun kullanımını ve zevkini arttırmamıza neden olucak

#### 3.1 Ödül Sistemi

##### 3.1.1 Coin Sistemi

Oyunumuzda yol üzerinde bulunan coinleri karakterimiz toplamaya çalışıyor. Toplanan coinler oyuncumuzun puanını arttırmasını sağlıyor. Böylelikle oyunda puan farkı yaratacak bir seçenek sunularak oyunumuza amaç yükleniliyor.



Şekil 3.1 : TL Coin

### 3.1.2 Mıknatıs Sistemi

Oyun üzerinde bulunan bir diğer ödülümüz ise mıknatıs. Karakterimiz yol üzerinde mıknatısları toplayarak öncelikle yol üzerindeki diğer coinleri karakterimizin üzerine çekerek toplama kolaylığı sağlıyor. Mıknatısın bir diğer getirisi olarak ise puanı önemli ölçüde arttırarak oyunumuza etki ediyor. Tabiki bu mıknatıslar oyun üzerinde diğer ödülümüze göre daha nadir oluyor. Oluşturduğumuz coinleri yol üzerinde



Şekil 3.2 : Mıknatıs

toplamaya çalışırken öncelikle mıknatıs kontrolü yapılıyor. Eğer oyuncumuz daha önceden mıknatıs almış ise coinler karakterimize oluşturduğumuz görünmeyen temas küpümüze doğru uçacak. Bu işlemi coine 10 birimlik mesafe kala gerçekleştiriyoruz.

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

public class coin : MonoBehaviour
{
    player player;
    Transform temas_kupu;
    float mesafe;

    void Start()
    {
        player = GameObject.Find("player").GetComponent<player>();
        temas_kupu = GameObject.Find("player/temas_kupu").transform;
    }

    // Update is called once per frame
    void Update()
    {
        if(player.mıknatıs_alındı == true )
        {
            mesafe = Vector3.Distance(transform.position, temas_kupu.position);
            if(mesafe<=10)
            {
                transform.position = Vector3.MoveTowards(transform.position, temas_kupu.position, Time.deltaTime * 10.0f);
            }
        }
    }
}
```

Şekil 3.3 : Coin Mıknatıs İlişkisi Kod

### 3.1.3 Temas Küpü



Şekil 3.4 : Karakterimizin Temas Küpü

Karakterimizin üzerinde bulunan görünmeyen küp, mıknatıs alındığında coinlerin çekileceği bölgedir.

### 3.1.4 Puan Sistemi

Oyunumuzu oynayan kişilerin topladıkları coin, mıknatıslara göre ve ilerleme durumlarına göre oluşturulan bir sistemdir. Puan sistemi kişilerin kendisinin bir önceki rekorunu geçerek ya da başka kişilerle yarışmasına olanak sağlar. Oyunda toplanan coinler ekranın sağ köşesinde puanı ise sol köşesinde gösterilerek oyuncunun ilerleme durumunu anlaması hedeflenmiştir



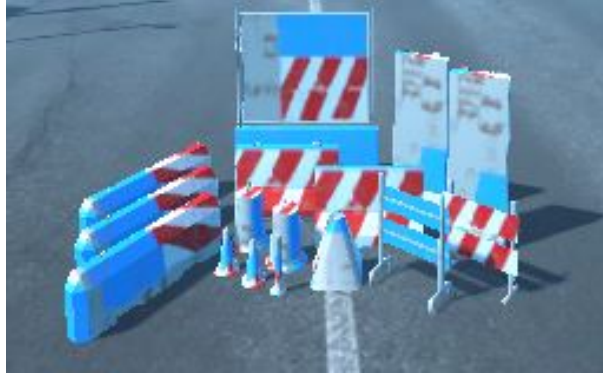
Şekil 3.5 : Oyun İlerleme Durum Ekranı

### 3.2 Engel Sistemi

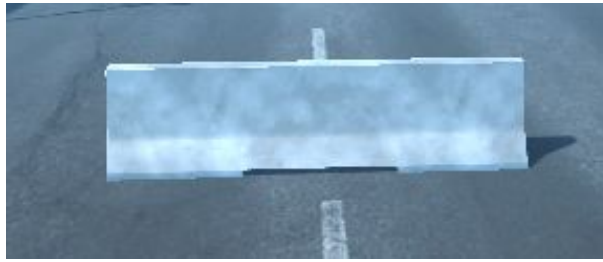
Oyunumuzda kullanıcımızın geçmesi gereken engeller bulunmaktadır. Bu engellerin üstünden zıplayarak veya yanından geçerek engelleri atlatması gerekir. Bu oyundaki ilerlemeyi zorlaştırarak oyun zevkini artırır. Oyunda iki tür engel çeşidi bulunmaktadır. Bunların ilki hareketli engel bir diğeri ise hareketsiz engellerdir.

#### 3.2.1 Hareketsiz Engel

Hareketsiz engel oyunumuzun yol üzerinde rastgele bölgelerde karakterimizin karşısına çıkan sabit engellerdir. Geçilmesi hareketli engellere göre daha kolaydır.



Şekil 3.6 : Yol Yapım Engelleri



Şekil 3.7 : Basit Beton Bariyer



Şekil 3.8 : Çöp Kutusu

### 3.2.2 Hareketli Engel

Hareketli engel oyunumuzun yol üzerinde rastgele bölgelerde karakterimizin karşısına çıkan ve üzerine doğru hareket eden engellerdir. Geçilmesi hareketsiz engellere göre daha zordur. Hareketli engeller üzerimize doğru gelirken korna sesi çıkararak öncü bir haber vermektedir. Hareketli engellerinin üzerine gelirken çıkaracağı sesin



Şekil 3.9 : Araba

kodlaması ise aşağıda gösterildiği gibi kodlanmıştır. Kodlamada görüldüğü üzere ses klipleri oluşturarak gerekli atama yapılmıştır. Aynı zamanda bir adet hız değişkeni



Şekil 3.10 : İtfaiye Kamyonu

belirlenmiştir. Bu hız engelin karaktere doğru hareketinin hızını belirler. Engellerin hangi koordinat üzerinde hareket edeceğide belirtilmiştir.

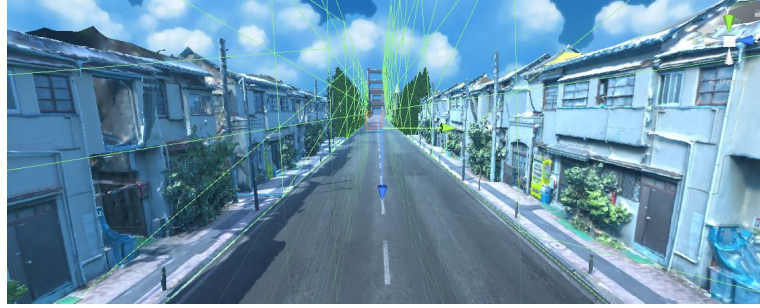
```
araba.cs x MenuManage.cs
Diğer Dosyalar
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class araba : MonoBehaviour
6 {
7     public AudioSource sesdosyasi2;
8     public AudioClip arabasesi2;
9     float hiz = 5;
10    void Update()
11    {
12        transform.Translate(0, 0, hiz * Time.deltaTime);
13        sesdosyasi2.PlayOneShot(arabasesi2);
14    }
15 }
16
```

Şekil 3.11 : Hareketli Engel Kod

## 4. OYUN KODLAMASI

### 4.1 Düşme Engeli

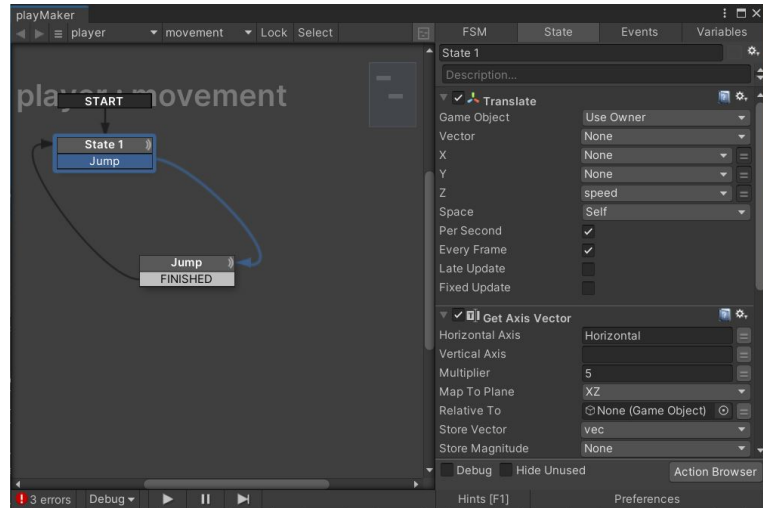
Oyunda koşan karakterimizin ilerlediği yol güzergahından düşmemesi için oluşturulan gizli engellerdir.



Şekil 4.1 : Düşme Engeli

### 4.2 Karakter Hareket Sistemi

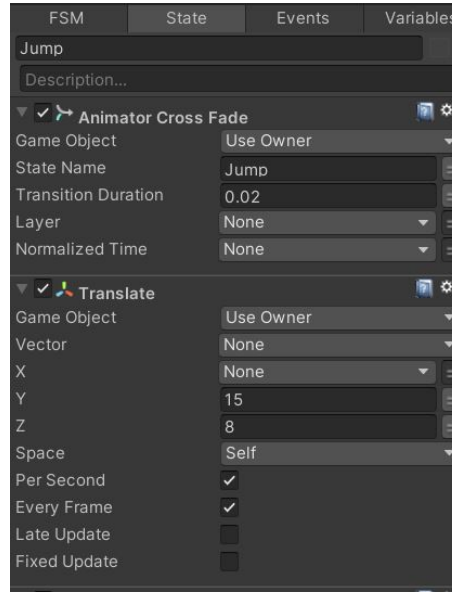
Karakterimizin yol üzerinde istediği konuma doğru hareket etmesini sağlayan ve hangi hızda ilerlemesi belirlediğimiz kod. Bu kodumuz Unity de kolaylık sağlayan Playmaker teknolojisiyle yazılmıştır. Bu sayede ekstra kod kalabalığından kurtulmuş bulunmaktayız.



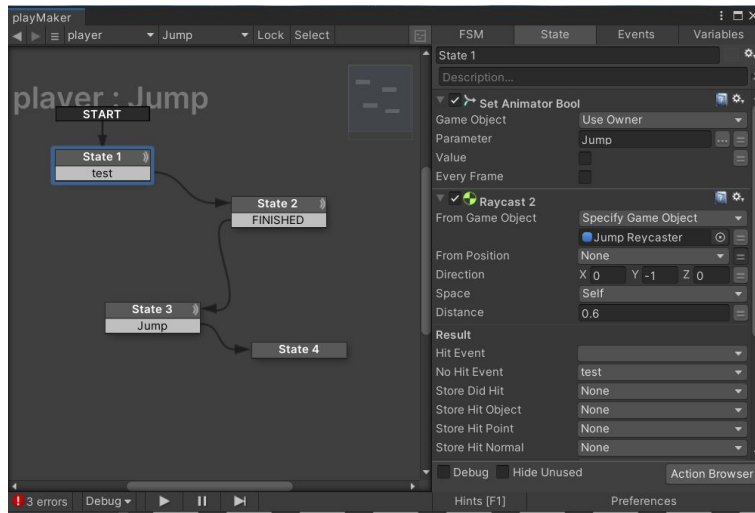
Şekil 4.2 : Karakter Hareket Kodu

### 4.2.1 Zıplama Sistemi

Karakterimizin Hareket ederken zıplamasına olanak sağlayan kodumuz. Bu kodda Playmaker teknolojisiyle yazılmıştır. Zıplama kodunda öncelikle kısa bir süre içinde animasyona geçmesi ve belirli yükseklikte hangi yöne doğru zıplayacağı belirlenerek ona göre karakterin vücut hareketlerinin değişikliği atanmıştır.



Şekil 4.3 : Karakter Hareket Kodu

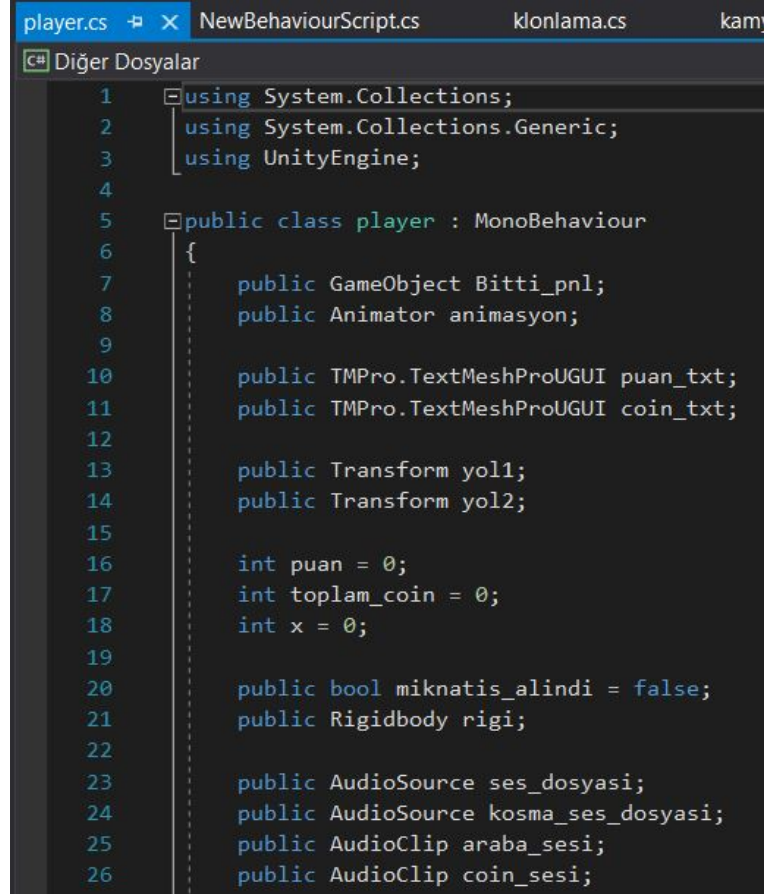


Şekil 4.4 : Karakter Hareket Kodu



### 4.3 Karakter Kodlaması

Karakterimize gerekli atamaları yapmamızı sağlamak için öncelikle değişken ve game objectlerimizi tanımlıyoruz. Şekil 4.5 de görüleceği üzere öncelikle bitiş ekranı



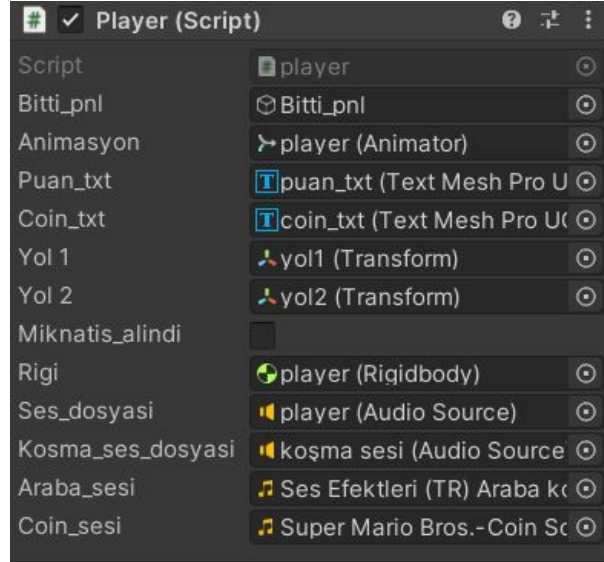
```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class player : MonoBehaviour
6 {
7     public GameObject Bitti_pnl;
8     public Animator animasyon;
9
10    public TMPro.TextMeshProUGUI puan_txt;
11    public TMPro.TextMeshProUGUI coin_txt;
12
13    public Transform yol1;
14    public Transform yol2;
15
16    int puan = 0;
17    int toplam_coin = 0;
18    int x = 0;
19
20    public bool miknatis_alindi = false;
21    public Rigidbody rigi;
22
23    public AudioSource ses_dosyasi;
24    public AudioSource kosma_ses_dosyasi;
25    public AudioClip araba_sesi;
26    public AudioClip coin_sesi;
```

Şekil 4.5 : Karakter Kodu

paneli ve animatorümüzün çalışması için gerekli animatör tanımlanmıştır. Daha sonra puan ve toplanılan coinimizin ekrana yansıtılıp tutulması için textler tanımlanmıştır. Oluşturulan tasarımın iki parça haline getirerek yol1 ve yol2 olarak adlandırılmıştır. Son olarak ise ses dosyalarımızın çalışması için audioclip tanımlanmıştır.

### 4.3.1 Karakter Değişken Ve Game Object Atamaları

Karakter kod dosyamızda oluşturduğumuz değişken ve game objectlerin Unity üzerinden atamalarını yaparak doğru biçimde çalışmasını sağlıyoruz.



Şekil 4.6 : Player Atamaları

### 4.3.2 Sonsuz Yol Döngüsü ve Çarpışma Sistemi

Oluşturduğumuz görsel tasarımları ve diğer objeleri iki game object olarak kaydettik. Kaydettiğimiz bu yolların üzerinde olan görünmez duvarlara çarpıştığı zamana göre birini diğerinin önüne geçiriyoruz. Aynı zamanda ekleme için gerekli koordinatları belirlendi. Yol üzerinde çarpıştığımız objenin engel olması durumunda oyunu durdurarak bitiş panelini çalıştırıyor. Eğer objemiz coin ise gerekli ses dosyasını çalıştırarak puanımızı ve toplam coinimizi güncelliyor. Aldığımız nesne mıknaş ise ona göre yapılması gereken işlemleri yapıyor.

```

private void OnTriggerEnter(Collider other)
{
    if (other.gameObject.name == "SON_1")
    {
        yol2.position = new Vector3(yol1.position.x - 0.4f, yol1.position.y, yol1.position.z);
    }
    if (other.gameObject.name == "SON_2")
    {
        yol1.position = new Vector3(yol2.position.x - 377.0f, yol2.position.y, yol2.position.z);
    }

    if (other.gameObject.tag == "enemy")
    {
        animasyon.SetBool("killed1", true);
        Bitti_pnl.SetActive(true);
        kosma_ses_dosyasi.enabled = false;
        Time.timeScale = 0.0f;
    }
    if (other.gameObject.tag == "coin")
    {
        ses_dosyasi.PlayOneShot(coin_sesi);
        Destroy(other.gameObject);
        toplam_coin++;
        puan += 5;
        puan_txt.text = puan.ToString("00000");
        coin_txt.text = toplam_coin.ToString();
    }
    if (other.gameObject.tag == "mıknatis")
    {
        GameObject[] tum_mıknatislar = GameObject.FindGameObjectsWithTag("mıknatis");
        foreach (GameObject mıknatis in tum_mıknatislar)
        {
            Destroy(mıknatis);
        }
        mıknatis_alindi = true;
        Invoke("mıknatisi_resetle", 10.0f);
    }
}

```

Şekil 4.7 : Sonsuz Döngü Ve Çarpışmalar Kodu

#### 4.3.3 Zıplama Sesi Ve Mıknatıs Kontrol

Bu kod parçasında mıknatısımız alınmamış durumda ise mıknatısımızın bool değeri true iken false olarak güncelliyoruz. Bir diğer yaptığımız işlem ise karakterimizin yol ile çarpışmasını kontrol ederek koşup koşmadığını kontrol ediyoruz. Yani zıplamış durumda mı yoksa koşar durumdamı olması. Bu duruma göre koşma sesi efektini kesip devam ettiriyoruz.

```

void mıknatisi_resetle()
{
    mıknatis_alindi = false;
}

private void OnCollisionStay(Collision collision)
{
    kosma_ses_dosyasi.enabled = true;
}

private void OnCollisionExit(Collision collision)
{
    kosma_ses_dosyasi.enabled = false;
}

```

Şekil 4.8 : Mıknatıs Kontrol Ve Ses Kontrol Kodu

#### 4.3.4 Puan Artımı

Bu kod parçasında ise puanımızı Invoke komutu ile her saniyede bir arttırıyoruz.

```
void Start()
{
    InvokeRepeating("puanartisi", 0, 1);
}
void puanartisi()
{
    puan++;
    puan_txt.text = puan.ToString("00000");
}
```

Şekil 4.9 : Mıknatıs Kontrol Ve Ses Kontrol Kodu

#### 4.4 Engel Ve Ödül Klonlama

Oyunumuzda oluşacak engel ve ödülleri klonladığımız kod dosyamız. Bu kodumuz sayesinde sürekli yeni engeller oluşturarak ve silerek engel ve ödül sürekliliği sağlıyoruz.

##### 4.4.1 Klonlama Kodu

Öncelikle klonlayacağımız objeleri tanımlıyoruz. Daha sonra oluşturulacak klonların nerede oluşturcağımıza göre koordinat içeren değişkenlerimizi oluşturuyoruz. Daha sonra oyun boyu çalışacak oluşturduğumuz fonksiyonları belirliyoruz.

```
klonlama.cs kanyon.cs coin.cs bitti_pnl.cs araba.cs MenuManage.cs
Diğer Dosyalar
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class klonlama : MonoBehaviour
6 {
7     public GameObject tlcoin;
8     public GameObject araba;
9     public GameObject bariyer;
10    public GameObject bariyer2;
11    public GameObject cöpkutusu;
12    public GameObject kanyon;
13    public GameObject lastikengel;
14    public GameObject miknatis;
15
16    public Transform player;
17    float silinme_zamanı = 7.0f;
18    float sağ_z_koordinat = 3.0f;
19    float sol_z_koordinat = -3.0f;
20    float orta_koordinat = 0.0f;
21    int sayı = 0;
22    void Start()
23    {
24        InvokeRepeating("sayac", 0, 1.0f);
25
26        InvokeRepeating("nesne_klonla", 2, 4.0f);
27        InvokeRepeating("nesne_klonla2", 2, 10.0f);
28        InvokeRepeating("nesne_klonla", 120, 7.0f);
29        InvokeRepeating("nesne_klonla2", 180, 16.0f);
30
31    }
```

Şekil 4.10 : Klonlama Kod

#### 4.4.2 Klonlama Oluşturma

Bu kod parçasında ise iki adet fonksiyon oluşturduk. Bu fonksiyonların biri hareketli nesne diğeri ise hareketsiz nesne oluşturuyor. Bu oluşturmalar rastgele seçilerek gerekli aralıklarda yine rastgele oluşturulmuştur.

```
void nesne_klonla()
{
    int rastsayi = Random.Range(0, 100);

    if (rastsayi > 0 && rastsayi < 40)
    {
        klonla(bariyer, 0f);
    }
    if (rastsayi > 40 && rastsayi < 70)
    {
        klonla(bariyer2, 0f);
    }
    if (rastsayi > 70 && rastsayi < 100)
    {
        klonla(cöp kutusu, 1.0f);
    }
}

void nesne_klonla2()
{
    int rastsayi2 = Random.Range(0, 100);

    if (rastsayi2 > 0 && rastsayi2 < 40)
    {
        klonla2(araba, 1.0f);
    }

    if (rastsayi2 > 40 && rastsayi2 < 80)
    {
        klonla2(kamyon, 1.0f);
    }

    if (rastsayi2 > 80 && rastsayi2 < 100)
    {
        if (player.gameObject.GetComponent<player>().miknatis_alindi == false)
        {
            klonla2(miknatis, 1.0f);
        }
    }
}
```

Şekil 4.11 : Klon Oluşturma

#### 4.4.3 Tekli Klon Ve Konumu Belirleme

Bu klon oluşturma kodumuzda rastgele birer adet nesne oluşturma ve diğer kısımlarında ise coin oluşturma işlemi yapıyoruz. Bu nesnenin konumu ve yolun ikili ya da tek tarafında olacağı rastgele olarak belirleniyor. Daha sonra oyunda hafıza alanı kaplamamsı için oluşturulan nesne geçilince silme işlemi yapılıyor.

```

void klonla2(GameObject nesne2, float Y_koordinat2)
{
    GameObject y_klon = Instantiate(nesne2);
    GameObject y_klon2 = Instantiate(nesne2);

    int rast = Random.Range(0,100);
    if(rast < 25)
    {
        y_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, orta_koordinat);
    }
    if (rast > 25 && rast < 50)
    {
        y_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, sol_z_koordinat);
    }
    if (rast > 50 && rast < 70)
    {
        y_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, sag_z_koordinat);
    }
    if (rast > 70 && rast < 80)
    {
        y_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, sol_z_koordinat);
        y_klon2.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, sag_z_koordinat);
    }
    if (rast > 80 && rast < 90)
    {
        y_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, orta_koordinat);
        y_klon2.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, sol_z_koordinat);
    }
    if (rast > 90 && rast < 100)
    {
        y_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, orta_koordinat);
        y_klon2.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat2, sag_z_koordinat);
    }
    Destroy(y_klon, silinme_zamani);
    Destroy(y_klon2, silinme_zamani);
}

```

Şekil 4.12 : Klon Oluşturma Ve Konum

#### 4.4.4 Çoklu Klon Adedi Ve Konumu Belirleme

Bu kod parçasında ise oluşturulan nesneden çoklu bir biçimde yapıp rastgele olarak oluşturma konumu belirleniyor bu nesnelerde geçildikten belli bir süre sonra hafıza alanı kaplamaması için siliniyor.Böylelikle oyunumuzda kasma yapması engelleniyor.

```

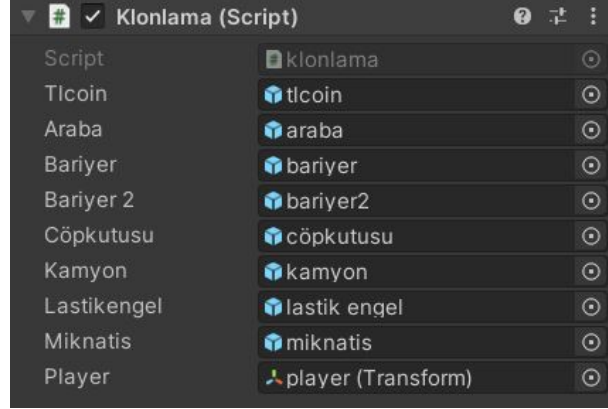
int rastsayi3 = Random.Range(0, 100);
int durum = Random.Range(0, 100);
if (durum < 40)
{
    if (rastsayi3 < 25)
    {
        yeni_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, orta_koordinat);
        yeni_klon2.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, orta_koordinat);
        yeni_klon3.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, orta_koordinat);
        yeni_klon7.transform.position = new Vector3(player.position.x - 20.0f, 1, sol_z_koordinat);
        yeni_klon8.transform.position = new Vector3(player.position.x - 22.0f, 1, sol_z_koordinat);
        yeni_klon9.transform.position = new Vector3(player.position.x - 24.0f, 1, sol_z_koordinat);
        yeni_klon10.transform.position = new Vector3(player.position.x - 20.0f, 1, sag_z_koordinat);
        yeni_klon11.transform.position = new Vector3(player.position.x - 22.0f, 1, sag_z_koordinat);
        yeni_klon12.transform.position = new Vector3(player.position.x - 24.0f, 1, sag_z_koordinat);
    }
    if (25 < rastsayi3 && rastsayi3 < 50)
    {
        yeni_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, sol_z_koordinat);
        yeni_klon2.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, sol_z_koordinat);
        yeni_klon3.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, sol_z_koordinat);
        yeni_klon4.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, sag_z_koordinat);
        yeni_klon5.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, sag_z_koordinat);
        yeni_klon6.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, sag_z_koordinat);
        yeni_klon7.transform.position = new Vector3(player.position.x - 20.0f, 1, orta_koordinat);
        yeni_klon8.transform.position = new Vector3(player.position.x - 22.0f, 1, orta_koordinat);
        yeni_klon9.transform.position = new Vector3(player.position.x - 24.0f, 1, orta_koordinat);
    }
    if (rastsayi3 > 50 && rastsayi3 < 75)
    {
        yeni_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, orta_koordinat);
        yeni_klon2.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, orta_koordinat);
        yeni_klon3.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, orta_koordinat);
        yeni_klon4.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, sol_z_koordinat);
        yeni_klon5.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, sol_z_koordinat);
        yeni_klon6.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, sol_z_koordinat);
        yeni_klon7.transform.position = new Vector3(player.position.x - 20.0f, 1, sag_z_koordinat);
        yeni_klon8.transform.position = new Vector3(player.position.x - 22.0f, 1, sag_z_koordinat);
        yeni_klon9.transform.position = new Vector3(player.position.x - 24.0f, 1, sag_z_koordinat);
    }
    if (rastsayi3 > 75 && rastsayi3 < 100)
    {
        yeni_klon.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, orta_koordinat);
        yeni_klon2.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, orta_koordinat);
        yeni_klon3.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, orta_koordinat);
        yeni_klon4.transform.position = new Vector3(player.position.x - 20.0f, Y_koordinat, sag_z_koordinat);
        yeni_klon5.transform.position = new Vector3(player.position.x - 22.0f, Y_koordinat, sag_z_koordinat);
        yeni_klon6.transform.position = new Vector3(player.position.x - 24.0f, Y_koordinat, sag_z_koordinat);
        yeni_klon7.transform.position = new Vector3(player.position.x - 20.0f, 1, sol_z_koordinat);
        yeni_klon8.transform.position = new Vector3(player.position.x - 22.0f, 1, sol_z_koordinat);
        yeni_klon9.transform.position = new Vector3(player.position.x - 24.0f, 1, sol_z_koordinat);
    }
}

```

Şekil 4.13 : Klon Oluşturma Ve Konum

#### 4.4.5 Klon Atamaları

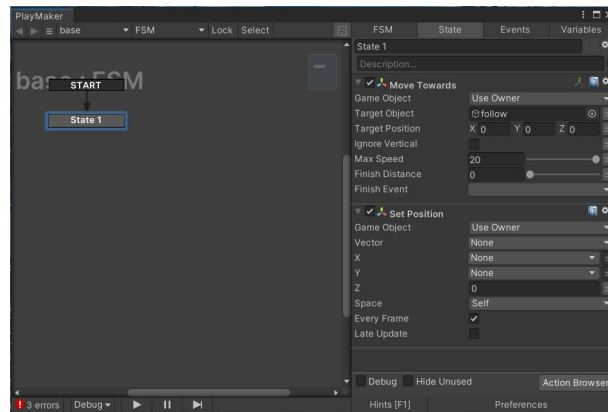
En son olarak belirlediğimiz game objectlerin Unity üzerinden tanımlamalarını yaparak doğru çalışmasını sağlıyoruz.



Şekil 4.14 : Klon Atamaları

#### 4.5 Kamera Takip Sistemi

Oyun boyu koşan karakterimizin konumu sürekli değişeceği için oynayan kişinin göreceği ortam sürekli karakterle beraber ilerlemelidir. Bu yüzden kameramızı karakterimizi baş kısmında oluşturduğumuz objeyi takip edecek şekilde Playmaker teknolojisiyle ayarladık.



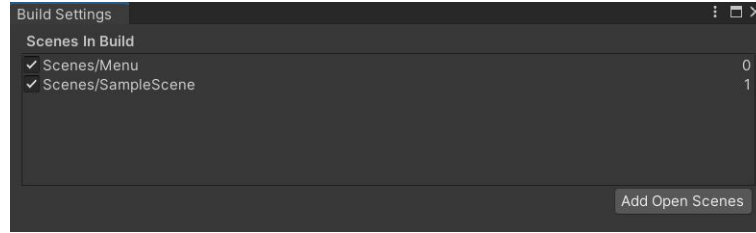
Şekil 4.15 : Kamera Takip Kodu





## 5. MENÜ VE BİTİŞ EKRANI

Oyunumuzda mobil platform için bir başlangıç menü ekranı ve bitiş ekranı olmalı bunun için yeni bir sahne oluşturularak gerekli görsel düzenlemeler ve kodlar yazılmıştır



Şekil 5.1 : Sahneler

### 5.1 Menü Oluşturma Kodu

Oluşturduğumuz kod ile menümüzde bulunan ilgili tuşlara basılarak oyunun başlaması ve çıkış sağlamasını yapıyoruz.

```
MenuManage.cs
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class MenuManage : MonoBehaviour
7 {
8
9     public void startButton()
10     {
11         Time.timeScale = 1.0f;
12         SceneManager.LoadScene("Scenes/SampleScene");
13     }
14     public void quit()
15     {
16         Application.Quit();
17     }
18 }
19
20
```

Şekil 5.2 : Menü Kod

### 5.1.1 Menü Görsel Görünüm

Aşağıda ise menü görünümü ve oradan açılan ayarlar görünümü yer almaktadır.



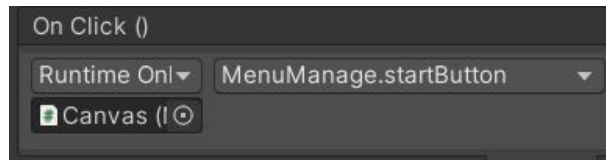
Şekil 5.3 : Menü



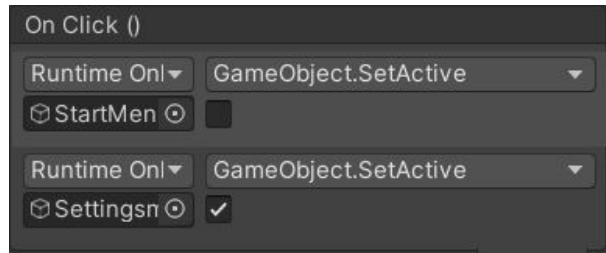
Şekil 5.4 : Menü Ayarlar Kısımı

### 5.1.2 Menü Button Atamaları

Menümüzde gerekli butonların atamalarıyla tıklandığında ne yapılacağını belirlemek adına yazılan kod.



Şekil 5.5 : Buton Atama



Şekil 5.6 : Buton Atama

## 5.2 Bitiş Panel Kod

Oyunda karakterimiz herhangi bir engele çarpması sonucu yanma durumunda oyunu bitirilmesi gerekiyor bu yüzden de oyuna bir bitiş paneli aylanmıştır. Kod aşağıdaki gibidir. Bu butonlara basılması durumunda yapılacak işlemler kodlanmıştır.

```
bitti_pnl.cs araba.cs MenuManage.cs
Diğer Dosyalar
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class bitti_pnl : MonoBehaviour
7 {
8     public void buton1()
9     {
10         Time.timeScale = 1.0f;
11         SceneManager.LoadScene("Scenes/SampleScene");
12     }
13     public void buton3()
14     {
15         Application.Quit();
16     }
17
18
19
20
21
```

Şekil 5.7 : Bitiş Paneli Kodu

### 5.2.1 Bitiş Paneli Görünüm

Aşağıda ise bitiş panelimizin görünümü bulunmaktadır.



Şekil 5.8 : Bitiş Paneli Görünümü



## 6. KULLANILAN TEKNOLOJİLER

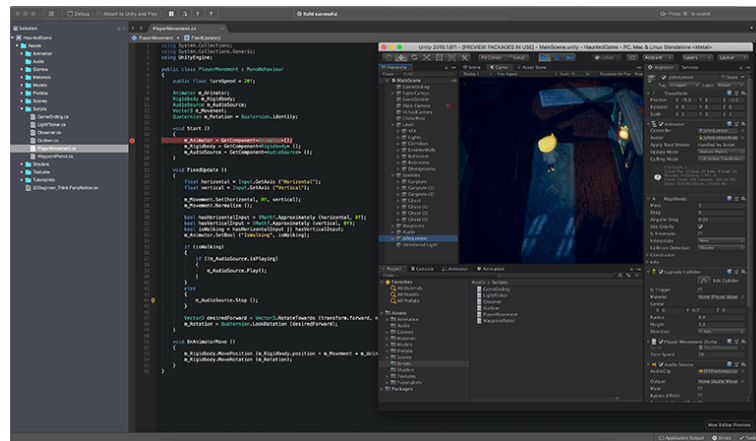
### 6.1 Visual Studio

Visual Studio, birçok programlama dilini kullanarak program, uygulama ya da web sitesi yapabileceğiniz bir IDE yani entegre geliştirme ortamıdır. Microsoft Windows için bilgisayar programları, web siteleri, web uygulamaları, web hizmetleri ve mobil uygulamalar geliştirmek için kullanılır.

Visual Studio, Windows API, Windows Forms, Windows Presentation Foundation, Windows Store ve Microsoft Silverlight gibi Microsoft yazılım geliştirme platformlarını kullanır. Hem yerel kod hem de yönetilen kod üretebilir.

Visual Studio, IntelliSense'i (kod tamamlama bileşeni) ve kod yeniden düzenleme işlemini destekleyen bir kod düzenleyici içerir. Entegre hata ayıklayıcı, hem kaynak düzeyinde hata ayıklayıcı hem de makine düzeyinde hata ayıklayıcı olarak çalışır. Diğer yerleşik araçlar arasında bir kod profili oluşturucu, GUI uygulamaları oluşturmak için form tasarımcısı, web tasarımcısı, sınıf tasarımcısı ve veritabanı şeması tasarımcısı bulunur. Neredeyse her düzeyde işlevselliği artıran eklentileri kabul eder.

Visual Studio'yu unity de yazacağım kod için editör olarak kullanacağım.



Şekil 6.1 : Visual Studio

Visual Studio, farklı programlama dillerini destekler ve dile özgü bir hizmet olması koşuluyla, kod düzenleyicisinin ve hata ayıklayıcının hemen hemen tüm programlama dillerini desteklemesine olanak tanır. Yerleşik diller arasında C, C ++ ve C ++ / CLI (Visual C ++ ile), VB.NET (Visual Basic .NET ile), C (Visual C ile), F (Visual Studio 2010'dan itibaren) ve TypeScript (Visual Studio 2013 Update 2'den itibaren) bulunur.

Microsoft, eklentileri destekleyen ve ücretsiz olarak kullanılabilen Community sürümü, Visual Studio'nun ücretsiz bir sürümünü sunar. Visual Studio'yu buradan indirebilir ve hemen kullanmaya başlayabilirsiniz.

## 6.2 Unity

Unity 3D Rusyada geliştirilen ve yayınlanan bir oyun motorudur. Nedir ne değildir ayrıntılı ele alacağız. Fakat şunu da belirtelim ki Öncelikle farklı bir amaç için üretilmiş olup daha sonra oyun yapmak amacıyla kullanılmış ve geliştirilmiştir. Unity'nin yazılmasının sebebi oyunları bilgisayara yüklemekten Unity Web Player ile internet üzerinden oynayabilmektir. İlk bu fikir gerçekten değer görmüştür. Çünkü oyunlarda korsanın önüne geçilecek ve oyun sadece kendi internet sitesinden oynatılabilecektir. Fakat Unity 3D bununla yetinmeyerek kendini daha da geliştirdi.

Unity 3D 2012 aralık ayında yayımlanan 4.0 versiyonu ile resmi olarak Desktop, Android, IOS, Flash Player, PS3, Web Player ve XBOX platformları için oyun yapmanıza imkân sağlamaktadır.

Yani ios ve android için ayrı ayrı oyun geliştirmek zorunda kalmazsınız. Her ikisi için de oyununuzu export edebilirsiniz.

Unity 3D oyun motoru ile hem 2 boyutlu hem de 3 boyutlu oyunlar yapılabilir.

Unity 3D oyun motoru Unity Engines tarafından C/C++ ile geliştirilmiştir.



Şekil 6.2 : Unity

Diğer motorlardan farkı hem ucuz olması, hem kaynağının bol olması, hemde yaptığınız oyunu bir tıklamayla bütün platformlara çevirebiliyor olması en büyük avantajlarından. Android için yaptığınız bir oyunu ios gibi platformlarda çevirmenizi sağlar bu sayede bir daha aynı oyun için diğer platformlarda uğraşmanız gerekmez. Bunun için çok fazla tercih edilen bir oyun motorudur.

### 6.3 Unity Asset Store

Unity Asset Store Unity 'nin eklenti indirme ve satın alma bölümüdür. Burada bir çok eklentiye ücretsiz olarak ulaşabiliriz. Burdan bazı ortam şekillerini ve eklentileri almayı düşünüyorum.



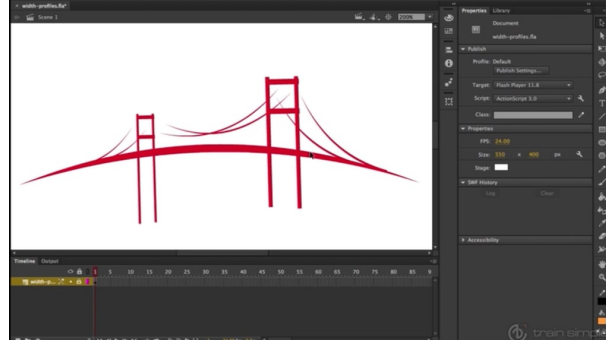
Şekil 6.3 : Unity Asset Store

### 6.4 Adobe Animate

Oyunlar, televizyon programları ve web için etkileşimli animasyonlar tasarlayın. Karikatürlere ve bant reklamlara hayat verin. Animasyonlu karalamalar ve avatarlar oluşturun. Ayrıca e-öğrenme içeriklerine ve infografiklere hareket katın. Animate ile birden çok platformda hemen hemen her formatta hızla yayın yapabilir ve izleyicilere herhangi bir ekrandan ulaşabilirsiniz.

Güçlü illüstrasyonlar ve animasyon araçları kullanarak oyunlar için etkileşimli web ve mobil içerikler oluşturun. Oyun ortamları hazırlayın, başlangıç ekranları tasarlayın ve bunları seslerle bütünleştirin. Animasyonlarınızı genişletilmiş gerçeklik deneyimleri olarak paylaşın. Animate ile varlık tasarımı ve kodlama işlemlerinizin tamamını doğrudan uygulamanın içinde gerçekleştirebilirsiniz.

Gerçek fırça hissi veren Adobe Fresco canlı fırçalarıyla daha etkileyici karakter eskizleri hazırlayıp çizin. Basit kare kare animasyonla karakterlerinize göz kırptırın,

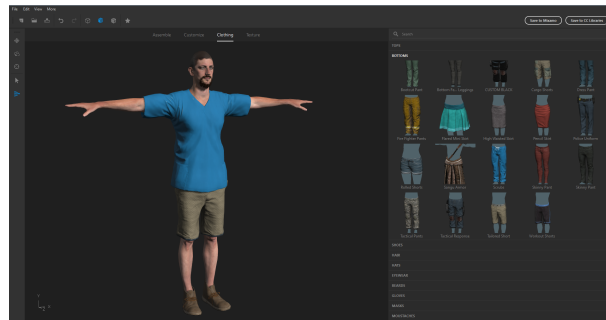


Şekil 6.4 : Adobe Animate

konuşturun ve yürütün. Hatta fare hareketi, dokunma ve tıklatma gibi kullanıcı etkileşimlerine yanıt veren etkileşimli web reklam bantları oluşturun.

## 6.5 Adobe Fuse

Adobe Fuse ile herhangi bir 3D veya animasyon bilgisine gerek olmadan oldukça detaylı ve tamamıyla sizin istediğiniz özelliklerde olacak 3D animasyonlu karakterler oluşturabiliyorsunuz. karakter oluştururken Adobe'nin sunduğu seçenekler oldukça fazla. Animasyon veya 3D terminolojisinde "rigging" denilen 3D modele hareketli iskelet giydirme işi bir kaç tıklamayla son derece kolay bir şekilde halledilebiliyor. Sonrasında rig monte edilen 3D model , animasyon olarak canlandırılabilir. Hareket seçenekleri de bir hayli çeşitli olarak sunulmuş. İşin bir diğer ilginç yanı 3d taramayla elde edilen verilerin de Adobe Fuse'a aktarılabilir olması. Yani isterseniz kendinizin bir 3D taramasını aldıktan sonra animasyonlu karakterinizi yaratabilirsiniz.



Şekil 6.5 : Adobe Fuse



## 7. UYGULAMADA YAPILABİLECEK GELİŞTİRMELER

- Oyun içeriğindeki görsel tasarımların İstanbul şehri ile alakalı olması planlanıyor. İleriki yerlerde veya versiyonlarda yeni şehirler eklenebilir
- Oyun içeriğine bilgi soruları kaynaklı yeni bir ek özellik konulabilir. Oyun içinde tercihen alınacak bu içerik sorduğu soruyu bilmesi karşılığında karakterimize ekstra özellik verebilir.
- Oyunumuza yeni bir can sistemi getirilerek yanma hakkı 3 yapılabilir. Bu canımız azaldığında ise yol üzerinde çıkartarak tekrar can kazanma sistemi yapılabilir.
- Çift zıplama özelliği getirilerek yüksekliği diğer engellerden daha fazla olan engellerin üzerinden atlanabilir yapılabilir. Bu durumda yol üzerinden alınabilir bir ödül yapılabilir.
- Oyun içindeki karakterimizin yeni bir görünümünü satın alınabilme sistemi getirilebilir. Bu satın alma işlemi oyun içinde topladığımız coinlerle yapılabilir.



# Kaynakça

Unity Nedir, <https://www.technopat.net/sosyal/konu/unity-nedir-nasil-kullanilir-oyun>,  
Eriřim Tarihi: 2021-01-05

Mobil Oyun Yapımı, <https://www.udemy.com/course/mobiloyunyapimi>, Eriřim Tarihi:  
2021-01-06

Mobil Oyun Yapımı, <https://tr.bitdegree.org/tutorial/oyun-nasil-yapilir/>, Eriřim Tarihi:  
2021-01-06

Adobe, <https://www.adobe.com/tr/creativecloud.html>, Eriřim Tarihi: 2021-01-07

Adobe Edge, <https://sanalkurs.net/adobe-edge-animate-nedir-ne-ise-yarar-8223.html>,  
Eriřim Tarihi: 2021-01-07

Adobe Fuse, <https://www.minifabrikam.com/blog/adobe-fuse-ile-animasyon-karakteri-yaratmak>,  
Eriřim Tarihi: 2021-01-07

Visual Studio, <https://www.abakuskitap.com/blog/icerik/visual-studio-nedir>, Eriřim  
Tarihi: 2021-01-08

Mobil Oyun Yapımı, <https://sebergame.medium.com/unity-oyun-gelistirme-e993fe88cbde>,  
Eriřim Tarihi: 2021-01-08

Literatür Arařtırması, <https://www.guvenlioyna.org.tr/galeri-detay/subway-surfers-incelemesi>,  
Eriřim Tarihi: 2021-01-08

Tez Yapım Arařtırması, <https://www.academia.edu>, Eriřim Tarih: 2021-04-15

3D Obje Alımı, <https://free3d.com/tr/3d-models>, Eriřim Tarihi: 2021-04-20

Texture Seçimleri, <https://www.textures.com>, Eriřim Tarihi: 2021-04-25

Karakter Oluřturma, <https://www.mixamo.com/>, Eriřim Tarihi: 2021-04-17

3D Model, <https://sketchfab.com>, Eriřim Tarihi: 2021-05-03



## **ÖZGEÇMİŞ**

**Ad Soyad:** Caner Savak

**Doğum Tarihi ve Yeri:** 01/09/1998

**E-Posta:** caner.svk35@gmail.com

### **ÖĞRENİM DURUMU:**

- **Lise:** 2016, Oya Ali Osman Keçici Anadolu Öğretmen Lisesi
- **Lisans:** 2021, Pamukkale Üniversitesi, Bilgisayar Mühendisliği