BC0650 Bitirme Çalışması : Run Like Hell !

Hacettepe Üniversitesi

Ertan TURAN & Bartu YURTSEVEN

29 Mayıs 2020

Özet

Run Like Hell, prosedürel sahne oluşturma teknikleri yardımıyla sonsüz koşu türünde bir oyundur. Geliştirmesi Unity3D ortamında modellemesi ise Maya3D ortamında yapılmıştır. Oyun, soygundan kaçan hırsızlardan birinin polisten kaçış koşusunu konu alır ve kaçışı esnasında olanların kontrolünü oyuncuya verir.

İçindekiler

1 Genel Bakış **2**
- 1.1 Oyunun Konusu . . . 2
- 1.2 Oyun Hakkında . . . 2
  - 1.2.1 Oyun Türü . . . 2
  - 1.2.2 Platform . . . 2
  - 1.2.3 Hedef Kitle . . . 2
  - 1.2.4 Geliştirme Araçları . . . 2

2 Oyun Mekanikleri **3**
- 2.1 Sonsuz Koşu Mekaniği . . . 3
- 2.2 Altın ve Para Mekaniği . . . 3
- 2.3 Günlük Ödül Mekaniği . . . 4

3 İçerik **5**
- 3.1 Modeller . . . 5
  - 3.1.1 Patlayici . . . 5
  - 3.1.2 Binalar . . . 5
  - 3.1.3 Yollar . . . 8
  - 3.1.4 Karakter . . . 8
- 3.2 Arayüz . . . 9
  - 3.2.1 Giriş . . . 9
  - 3.2.2 Ana Menü . . . 9
- 3.3 Animasyon . . . 9
- 3.4 Görsel Efektler . . . 9
- 3.5 Kodlama . . . 10
  - 3.5.1 Optimizasyon . . . 10
  - 3.5.2 Para Yönetimi . . . 11
  - 3.5.3 Prosedürel Yol Oluşturma . . . 12

4 Oyun İçi Ekran Görüntüleri **14**

Bölüm 1

Genel Bakış

1.1 Oyunun Konusu

İki kafadar bir gün bir bankayı soymaya karar verirler. Bu iş için her hazırlıkları yaparlar. Ancak , o gün geldiğinde işler yolunda gitmez. İçlerinden birisi heyecan ve korkudan soygunu paraları almaya devam ettikleri esnada yarıda bırakıp heyecanla kaçmaya başlar.ve yalnız kalan diğer kafadar da yakalanmamak için panikleyip arkadaşının arkasından koşmaya başlar. İki kafadarın kaçtığı esnada polis olay yerine varmak üzeredir. Bu esnada soyguncuların kaçtığını gören polis onları kovalamaya başlar. Önden giden kafadar koştuğu esnada paraları yola saçarak polisin dikkatini dağıtmaya çalışır. Arkadan giden para delisi diğer kafadar ise polisten kaçarken dahi paraları düşünmektedir ve arkadaşının paraları etrafa saçması hiç hoşuna gitmez ve polisten kaçarken arkadaşının yere düşürdüğü para ve altınları toplamaya çalışmaktadır. Tam olarak burada oyuncumuzun kafadarımıza saçlıan para ve altınları toplamada yardım edip para ile hayallerine ulaşmasında yardımcı olmasını bekliyoruz !.

1.2 Oyun Hakkında

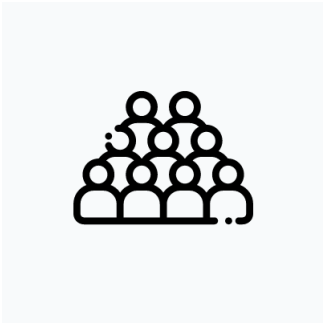
1.2.1 Oyun Türü

3 Boyutlu görsel sisteme sahip olan oyun tek oyunculudur. Tür olarak hyper-casual ve casual türlerine dahil edilebilir.

1.2.2 Platform

Bu oyun mobil (Android & iOS) platformuna uygun olarak tasarlandı

1.2.3 Hedef Kitle



Bu oyunun içeriği her yaştan insana hitap etmektedir. Oyunda çocukları veya yetişkineri rahatsız edecek içerik bulunmamaktadır. Macera ve casual tarzında oyunlarını seven 7 yaş ve üzeri oyuncu kitlesi için uygundur.

1.2.4 Geliştirme Araçları

- Unity 3D
- Adobe Photoshop
- Adobe Illustrator
- Maya

Bölüm 2

Oyun Mekanikleri

2.1 Sonsuz Koşu Mekaniği

Oyuncu prosedürel olarak oluşan harita üzerinde sonsuz bir koşu döngüsü içerisinde kendisini bulur. Koşmasının temel sebebi onu kovalayan polislerden kaçmaktır. Oyuncu koşu esnasında para ve altınlar toplayarak servetini arttıracaktır. Bu serveti daha sonra daha farklı karakterler ve kıyafetler alabilmek için kullanacaktır. Kafadarımızın karşısına koşu esasında engeller çıkacaktır. Bu engeller. Polis barikatı, yol çalışması, çöp kutusu vb. gibi yolda bulunabilecek olan engeller olacaktır.Bu engellerin bir kısmından sağa veya sola manevra yaparak kurtulabilecek olmasına karşın , bir kısmından da kurtulabilmesi için karşısındaki engeli yok etmesi gerekecektir. Bunu yapabilmesinin tek yolu ise koştuğu esnada karşısına çıkan patlayıcıları da envanterine eklemesi olacaktır. Eğer oyuncu hiçbir patlayıcı elde edemediyse ve karşısına atlayarak veya sağa sola kaçarak kurtulayamayacağı bir engel çıkarsa kafadarımız yakalanacaktır ancak oyuncumuz patlayıcılar hariç servetini kaybetmeyeektir.

Bunun yanısıra kafadarımızın karşısına polisler veya yolda onu durdurmak isteyecek kişiler çıkacaktır. Bu insanları atlatabilmek için kafadarımızın bu insanlara yaklaştığı esanada yumruk atması gerekmektedir. Atlattığı her engelde geçtiği her kişide kafadarımız servetini toplayabilmek için daha fazla zamana sahip olacaktır.

Koşu esnasında , oyuncumuzun elde ettiği ilerlemye göre kazanacağı paralar da değişecektir. Koşunun başlarında yerde daha az para, altın ve patlayıcı karşısına çıkacaktır. İlerledikçe toplayabileceği şeylerin sayısı arttıkça değeri de karşısına çıkacak engellerin sayısı ve zorluğuyla orantılı olarak artacaktır.

2.2 Altın ve Para Mekaniği

Oyuncumuz, oyunda kazandığı altınları yalnızca markette harcayabilecektir. Ancak dolarları ise geliştirmeler almak için kullanabilecektir. Oyuncumuzun marketten alabileceği şeyler değişik renk ve bina kompozisyonuna sahip şehirler,kıyafetler ve aksesuarlarla sınırlı olacaktır.

Dolarları ise "Yükseltme(Upgrade)" ekranından servetindeki dolarları harcayarak elde eedeilbecektir. Kullanıcının satın alacağı yükseltmeler. Kafadarımızın çantasının kapasitesinin arttırılmasına yönelik olacaktır. Örneğin kullanıcının yaptığı yükseltmelere göre kafadarımızn çantasına koyabileceği patlayıcı sayısı 1 den 2 ye çıkabilir. Ya da toplayabileceği dolar ve altın miktarı da aynı şekilde arttırılabilir olacaktır.

Kullanıcı implemente edilecek olan ödeme sistemiyle altın alabilecek ancak dolar alamayacaktır. Bu sistem ile kullanıcı yalnızca oyundan alacağı keyfi arttırmaya yönelik harcama yapabilecektir. Oyunda ilerleme kaydedebilmesi için ise oyunu oynaması gerekmektedir. Oyunumuzda "Pay2Win" mantığı yerine "Pay2Fun" mantığını benimsedik.

2.3 Günlük Ödül Mekaniği

Oyuncumuz oyuna düzenliği olarak girdiği her gün için ekstra ödüller alacaktır. Örneğin ilk gün 100 dolar 2.gün 500 dolar 3. gün 1 patlayıcı 4.gün 1 altın gibi ilerleyerek artacak bir günlük ödül sistemi bulunacaktır. Bu ödül sistemi her ay prosedürel olarak yenilenir.

Bölüm 3

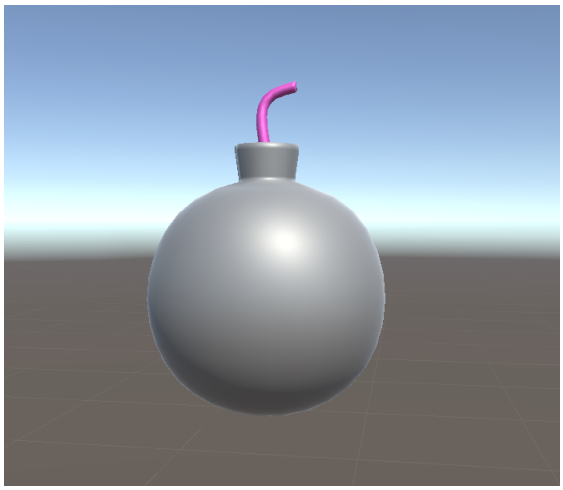
İçerik

3.1 Modeller

Oyunumuzda kullanılan tüm 3B modeller tamamiyle mobil uyumlu olacak şekilde tasarlanmıştır. Poligon ve tris sayıları draw cal yükleri ve mobilde oluşabilecek render kapasitesi sıkıntıları gözetilmiştir.

3.1.1 Patlayici

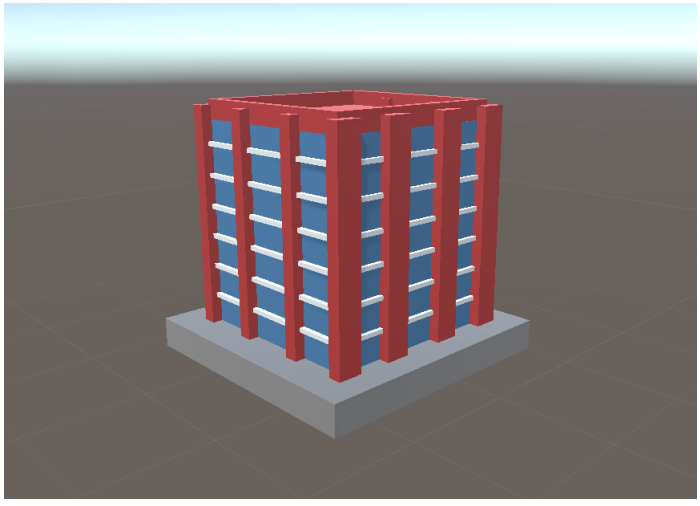
Bu bölümde oyunumuzda kullanılan el bombası modeli bulunmaktadır.



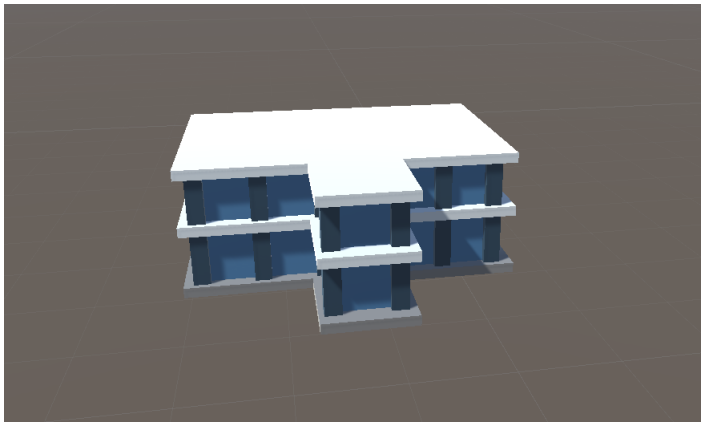
Şekil 3.1: El bombası

3.1.2 Binalar

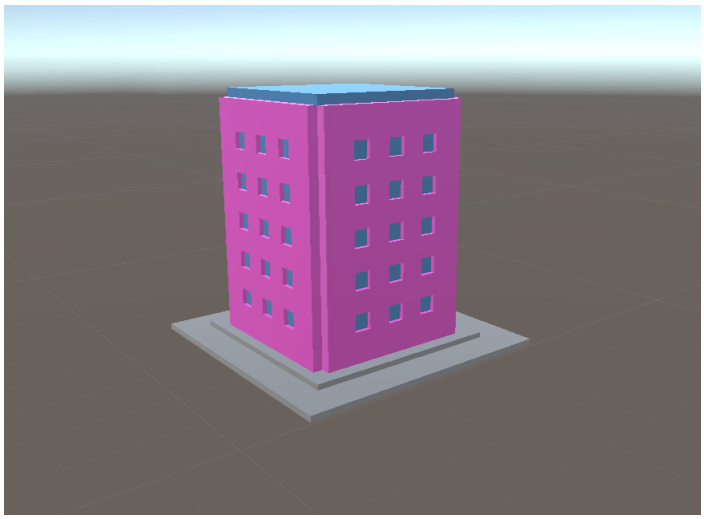
Bu bölümda oyunumuzda kullanılan mobil-uyumlu bine modelleri bulunmaktadır.



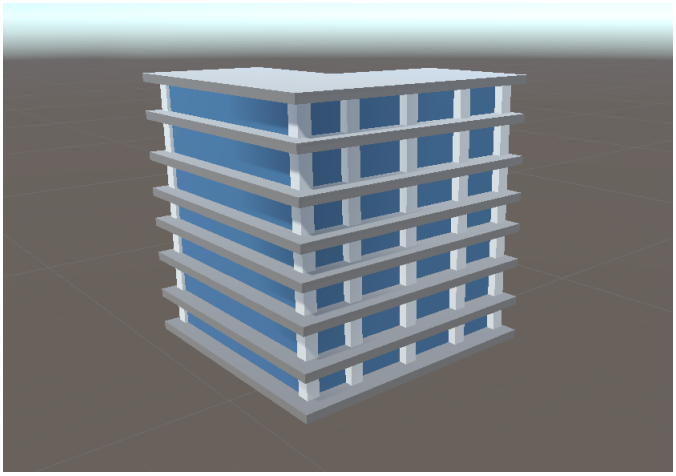
Şekil 3.2: Bina 1



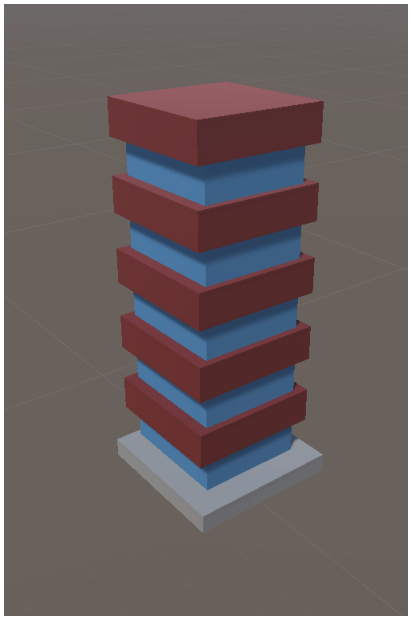
Şekil 3.3: Bina 2



Şekil 3.4: Bina 3

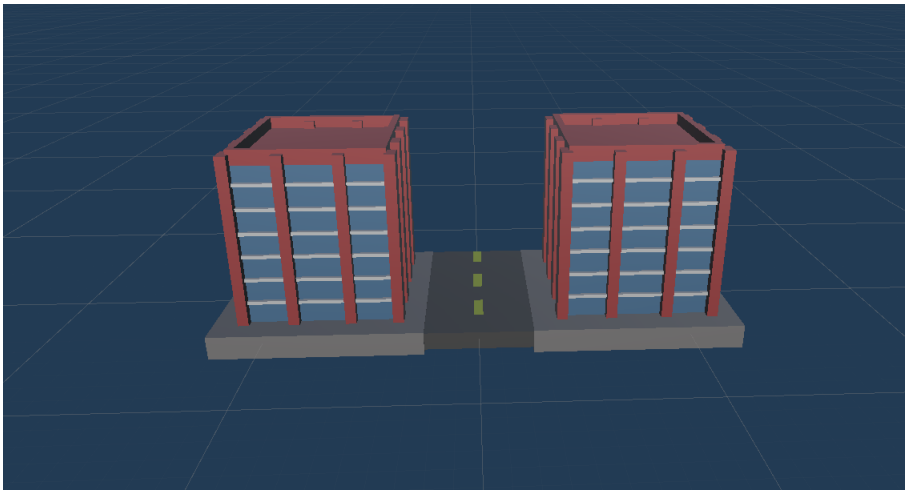


Şekil 3.5: Bna 4

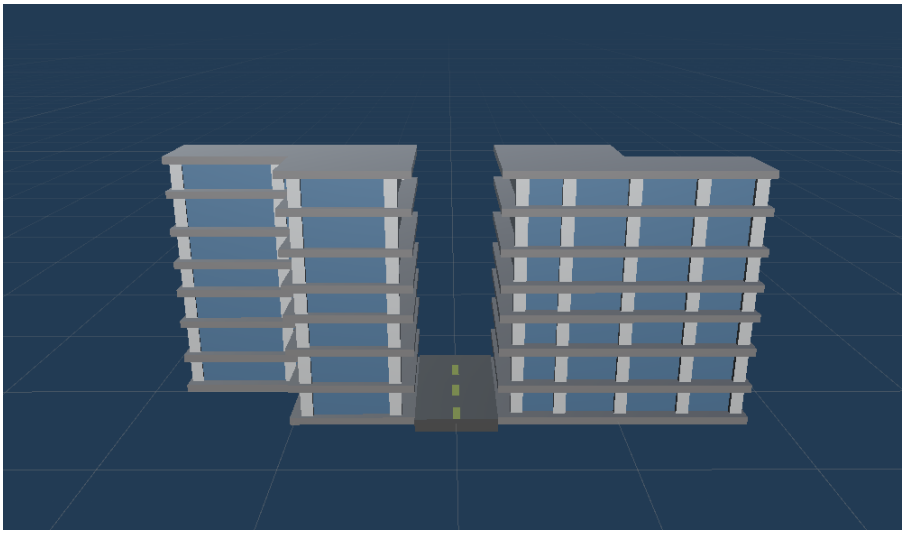


Şekil 3.6: Bina 5

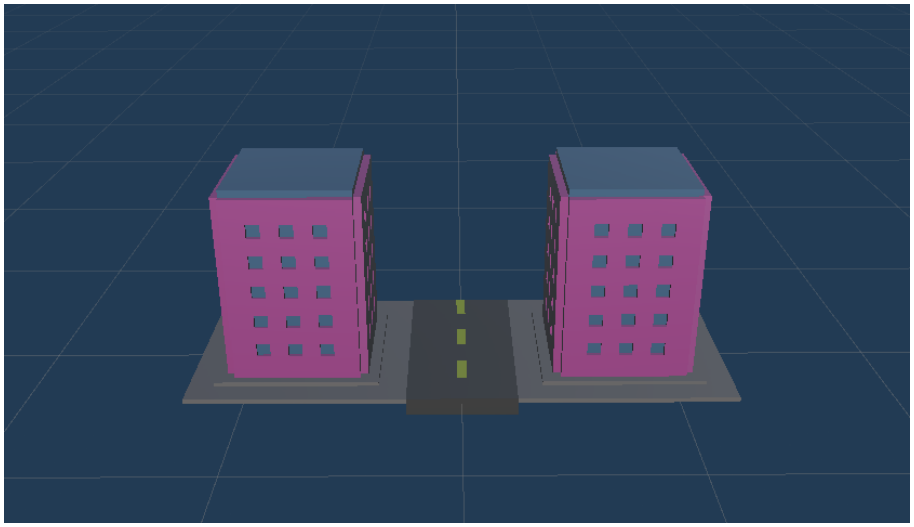
3.1.3 Yollar



Şekil 3.7: Yol 1



Şekil 3.8: Yol 2



Şekil 3.9: Yol 3

3.1.4 Karakter

Kafadarımız Sergei düz koşu ve engellerden sağa sola giden yada atlayabilen bir yapıya sahiptir. Polislerde arabalarıyla yada ahşap tahtalarla bariyer kurup bizi etkisiz hale getirmeye çalışırlar.

Şekil 3.10: Başlangıç karakteri

3.2 Arayüz

3.2.1 Giriş

- Splash screen gösterilir.
- (Oyun yüklenir.)
- Ana Menü gösterilir.

3.2.2 Ana Menü

- Yeni oyun: Oyun baştan başlatılır.
- Yükle: Daha önce kaydedilmiş olan bir oyuna devam edilir.
- Seçenekler: Ses ve görüntü ayarları yapılır.
- Günlük ödül alınır.
- Hakkında: Oyun geliştiricisi ve sürüm hakkında genel bilgiler sunar.
- Çıkış: Oyun kapatılır.

3.3 Animasyon

Tüm karakter aksiyonları düz koşu ve zıplama animasyonları olarak- gerçekleştirilir. Karakterin animasyonları arasında yumuşatılmış geçişler vardır. [1]

Diğer Objeler ile oyunda şehir similasyonu yaratılmaya çalışılmıştır. İki şeritli bir yol sağ ve sol tarafımızda modellenen 5 farklı çeşit bina yer almaktadır. Aynı zamanda yol kenarlarında trafik ışıkları ve trafik lambası bulunmaktadır.Yol uzerınde diğer soyguncu arkadaşımızın düşürdüğü paralar polisler ve engeller bulunmaktadır.s

3.4 Görsel Efektler

Oyuncunun yerden her para, altın veya patlayıcı toplayışında topladığı materyale uygun bir particle effect oynatılır. Particle effectler mobil uyumludur ve oyuncuya görsel keyif vermek maksadıyla kullanılmaktadır. Aynı zamanda günlük ödüllerini aldığı esnada da particle effect oynatılır. [2]

3.5 Kodlama

3.5.1 Optimizasyon

Object pool, dağıtık sistemlerde veya yazılım geliştirici tarafından yönetilmesi zor olan nesnelerde kullanılabilecek bir tasarım desenidir. Creational (Yaratıcı, nesnelerin oluşturulmasına yönelik) desenler içerisinde yer almaktadır. İstenilen nesnelerin sürekli olarak üretilmesi yerine, başlangıçta bir havuzu oluşturulur ve bu havuz nesneler ile doldurulur. Oyunumuzda sürekli oluşturulan neseneler (yollar,binalar ağaçlar, dolar ve coin gibi.) bu pool içerisinde tutularak sürekli döngü halinde kullanılmaktadır. [3]

```
public class ObjectPooler : MonoBehaviour
{

    public Dictionary<PooledObjectType, Queue<GameObject>> PoolDictionary;
    public List<PoolObjects> Pool;

    private Dictionary<PooledObjectType, int> _poolIndexes = new Dictionary<PooledObjectType, int>();
    private Dictionary<PooledObjectType, Transform> _poolMasters = new Dictionary<PooledObjectType, Transform>();

    public static ObjectPooler Instance;

    private void Awake()
    {
        Instance = this;
    }

    private void Start()
    {
        PoolDictionary = new Dictionary<PooledObjectType, Queue<GameObject>>();

        GameObject master = new GameObject( name: "Pool");

        for (int j = 0; j < Pool.Count; j++)
        {
            GameObject poolSpecifiMaster = new GameObject( name: Pool[j].Tag.ToString());
            poolSpecifiMaster.transform.parent = master.transform;

            Queue<GameObject> objectPool = new Queue<GameObject>();
            _poolIndexes.Add(Pool[j].Tag, j);

            _poolMasters.Add(Pool[j].Tag, poolSpecifiMaster.transform);


            for (int i = 0; i < Pool[j].Size; i++)
            {
                GameObject obj = Instantiate(Pool[j].Prefab);
                obj.transform.parent = poolSpecifiMaster.transform;
                IPooledObject iPool = obj.GetComponent<IPooledObject>();
                if (iPool == null)
                {
                    PooledObject temp = obj.AddComponent<PooledObject>();
                    iPool = temp;
                }
                iPool.PoolType = Pool[j].Tag;


                obj.SetActive(false);
                objectPool.Enqueue(obj);
            }

            PoolDictionary.Add(Pool[j].Tag, objectPool);
        }
    }

    public GameObject SpawnFromPool(PooledObjectType tag, Vector3 pos, Quaternion rot, GameObject parent = null)...

    public void Despawn(GameObject obj)...
```

Şekil 3.11: Object Pooling System

Generic'ler tasarlandığımız interface, class, metod yada parametrelerin (argümanların) belirli bir tip için değil bir şablon yapısına uyan her tip için çalışmasını sağlayan bir yapıdır. Oyunumuzda sahnemizde yer alan class yapılarına bir yenisi eklenmek istendiğinde sürekli kod tasarım döngüsüne girmemek için kullanılmaktadır.

```
using UnityEngine;

public class Singleton<T> : MonoBehaviour where T : Component
{
    public bool DontDestroyOnLoading = false;

    public static T Instance
    {
        get
        {

            if (_instance == null)
            {
                _instance = FindObjectOfType<T>();
                if (_instance == null)
                {
                    GameObject obj = new GameObject(typeof(T).Name);
                    obj.AddComponent<T>();
                }
            }

            return _instance;

        }
    }

    private static T _instance;

    protected virtual void Awake()
    {
        if (_instance == null)
        {
            _instance = this as T;
            if (DontDestroyOnLoading)
                DontDestroyOnLoad(gameObject);

        }
        else
        {
            Destroy(gameObject);
        }
    }

}
```

Şekil 3.12: Generic Singleton in Unity

3.5.2 Para Yönetimi

Para yönetimi arkaplanında, kullanıcının koşarken yerden topladığı altın ve dolarları çantasına ekleyen ve bunlar ile harcama yapaibilmesini sağlayan yazılım altyapısı bulunmaktadır. Bu sistem para elde edildiğinde meydana gelen efektleri tetikleyen olay-bazlı (event-driven) bir yapıda kodlanmıştır.

```
using System;
using UnityEngine;
using Random = UnityEngine.Random;

public class DollarManager : Singleton<DollarManager>
{
    [Range(0, 100)]
    public float MoneySpawnRate = 100;


    public MoneyText MoneyText;


    public Action<Money> OnMoneySpawned;

    protected override void Awake()
    {
        base.Awake();


        OnMoneySpawned += MoneySpawned;
    }

    private void MoneySpawned(Money monnay)
    {

    }

    public void SpawnMoney(GameObject obj)
    {
        int rand = Random.Range(0, 100);
        bool addMoney = rand < MoneySpawnRate ? true : false;
        if (obj != null && addMoney)
        {

            Vector3 tempVector = obj.transform.position;
            tempVector.y += 1;

            GameObject spawnedObj = ObjectPooler.Instance.SpawnFromPool(
                tag: PooledObjectType.Money, tempVector, Quaternion.identity);
            spawnedObj.transform.SetParent(obj.transform);
        }

    }

    public void MoneyChange(Money monnaaaayy)
    {
        MoneyText.ChangeText(PlayerController.Instance.PlayerMoney);
    }



}
```

Şekil 3.13: Para yönetimi

3.5.3 Prosedürel Yol Oluşturma

Prosedürel yol oluşturma algoritmasında, eşit uzunluklarda ölçeklendirilmiş modellerin ardı sıra rastgele sırayla konulması ve bunlara bağlı olarak etrafta bulunan ağaç,trafik ışığı vb gibi nesnelerinde buraya algoritma tarafından yerleştirilmesiyle oluşturulmuş bir ekosistem meydana gelmektedir.

```
    private void SpawnPlatform()
    {
        for (int i = 0; i < PlatformSize; i++)
        {
            _randomIndex = Random.Range(0, PlatformObjects.Length);
            GameObject obj = null;
            obj = ObjectPooler.Instance.SpawnFromPool(PlatformObjects[_randomIndex], _dummy.position, _dummy.rotation);

            if (PlatformObjects[_randomIndex] == PooledObjectType.StairsUp)
            {
                _dummy.Translate(x:0, y:5, z:0);
            }
            else if (PlatformObjects[_randomIndex] == PooledObjectType.StairsDown)
            {
                _eulerVector.y = 180;

                _dummy.Translate(x:0, y:-5, z:0);
                obj.transform.Rotate(_eulerVector);
                obj.transform.position = _dummy.position;

            }
            else if (PlatformObjects[_randomIndex] == PooledObjectType.PlatformTSection)
            {
                int randomRotation = Random.Range(0, 2) < 1 ? 90 : -90;

                _eulerVector.y = randomRotation;


                _dummy.Rotate(_eulerVector);
                _dummy.transform.Translate(translation:Vector3.forward * -10);

            }

            _dummy.transform.Translate(translation:Vector3.forward * -10);

            DollarManager.Instance.SpawnMoney(obj);
        }

        _spawnOnce = false;
    }


    private void FixedUpdate()
    {
        _dummy.position += PlayerController.Instance.transform.forward * -0.1f;

        float distance = (transform.position - _dummy.position).magnitude;
        if (distance < 50 && _spawnOnce)
        {
            SpawnPlatform();

        }
        else
        {
            _spawnOnce = true;
        }

    }
}
```

Şekil 3.14: Prosedürel yol (path) oluşturma algoritması

Bölüm 4

Oyun İçi Ekran Görüntüleri

Şekil 4.1: Oyun İçi Ekran Görüntüsü

Şekil 4.2: Oyun İçi Ekran Görüntüsü

Şekil 4.3: Oyun İçi Ekran Görüntüsü

Şekil 4.4: Oyun İçi Ekran Görüntüsü

Bibliography

[1] Mixamo by Adobe. *Misamo Animation*. URL: hhttps://www.mixamo.com/.

[2] Synty Studios. *Simple FX - Cartoon Particles*. URL: https://assetstore.unity.com/packages/vfx/particles/simple-fx-cartoon-particles-67834. (accessed: 01.05.2020).

[3] Ertan Turan. *Runtime-extendible Object-pooler within unity*. URL: https://github.com/ertanturan/Unity-Object-Pooling.