by yselim

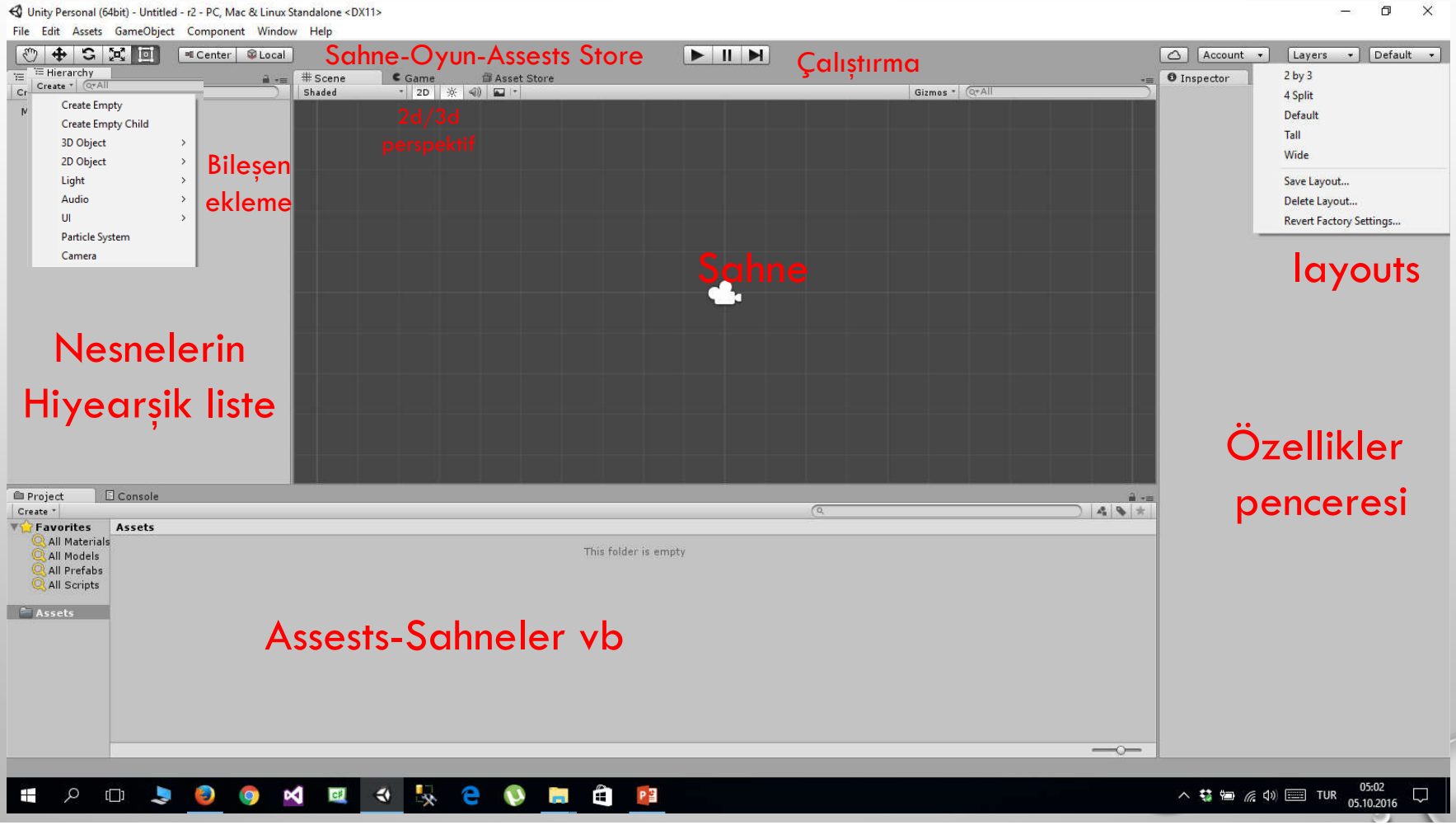
UNITY GÜNLÜĞÜ

ROLL-A-BALL ÖRNEĞİ (UPDATED)

GÜN 1

ROLL-A-BALL örneği

BAŞLAMADAN ÖNCE..

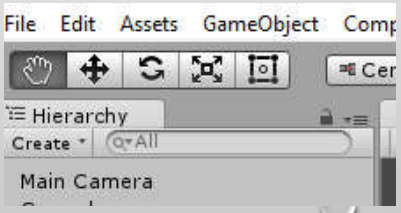


ROLL-A-BALL örneği

BAŞLAMADAN ÖNCE..

Bazı kısayollar.

- Sol Mouse butonu basılı tutulup nesneler seçilir hareket ettirilir.
- Ctrl + Sol buton : snap to grip hareket işlemi.
- Sağ Mouse buton basılı olarak bakış noktası referans; perpektif hareket
- Alt + Mouse sağ butonu ile obje /sahne referans; perspektif hareket
- Move / Scale / Rotate



ROLL-A-BALL örneği

1. Giriş
2. Çevrebirim ve oyuncu ayarları
    a) Oyunun Ayarları
    b) Oyuncuyu hareket ettirme
3. Kamera ve oyun alanı
    a) Kamerayı hareket ettirme
    b) Oyun alanı ayarları
4. Oyun skoru ve derleme
    a) Toplanabilir nesneler oluşturma
    b) Topları toplama
    c) Skoru gösterme
    d) Scene(sahne) değiştirme
    e) Oyunu derleyip çalıştırma

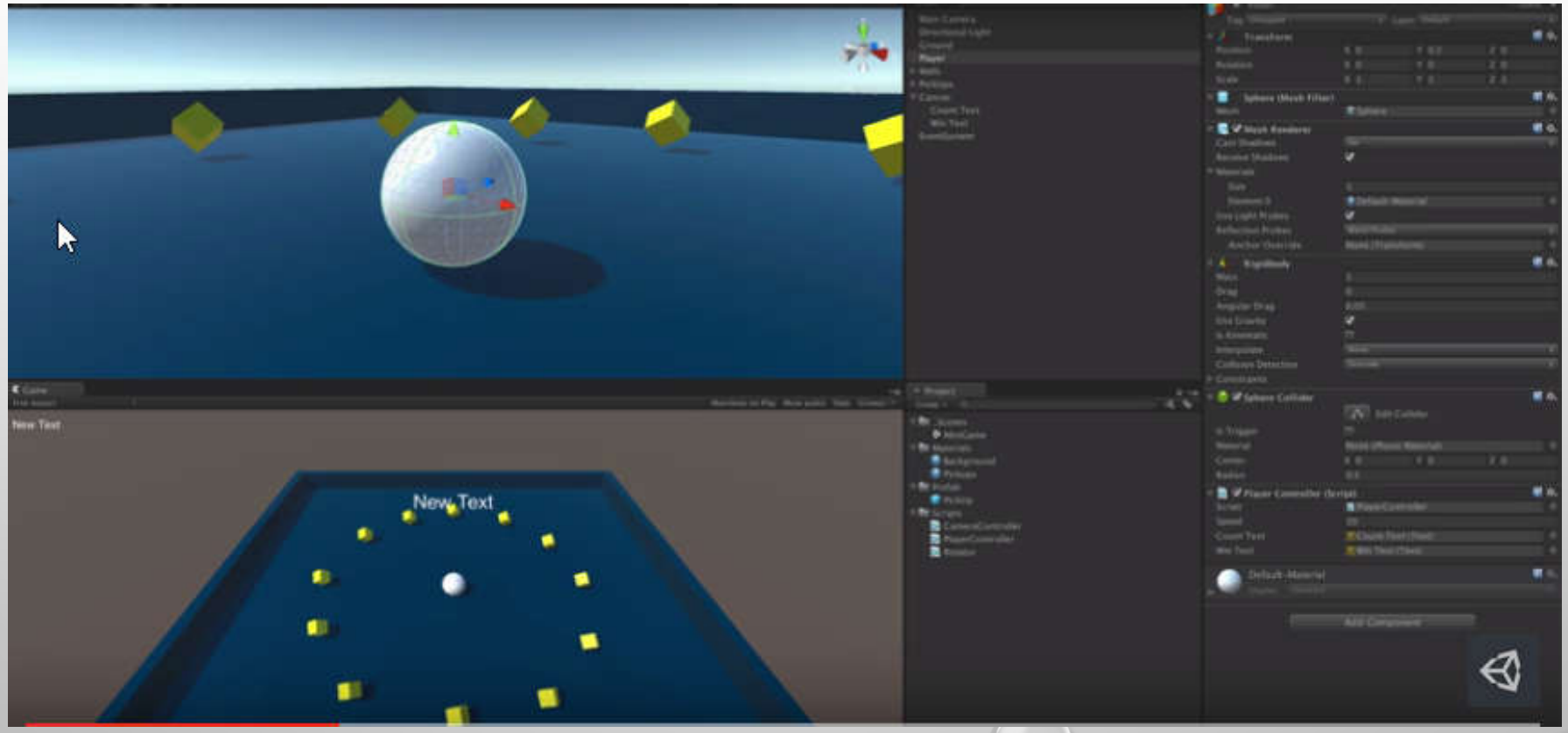
ROLL-A-BALL örneği

1. GİRİŞ

- Bu uygulamada UNITY kullanılarak temel primitiv şekillerin olduğu çeşitli nesnelerin toplanarak puan kazanıldığı basit bir top oyunu gerçekleştirilecektir.
- Oyun başlangıç seviyesinde temel konsept içermektedir. Yeni oyun nesneleri oluşturulacak, bu objelere bileşenler eklenecek. Sahnedeki (scene) bu bileşenler oyunun oluşturacaktır.
- Oyuncu oyun tahtası üzerindeki topu hareket ettirerek nesneleri toplayıp puan kazanacaktır. Kullanıcıdan gelen klavye girişlerine göre topa güç uygulayıp sahnede topu hareket ettireceğiz. Diğer nesnelerle topun kontak kurması halinde nesnelere toplanacak(collect) puan kazanılacaktır, bu puanlar skor textinde gösterilecektir.
- Bu projede her hangi bir assets, model, texture,sound,animation kullanılmayacaktır.

ROLL-A-BALL örneği

1. GİRİŞ (DEVAM)



ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI

Yeni bir proje oluşturalım.

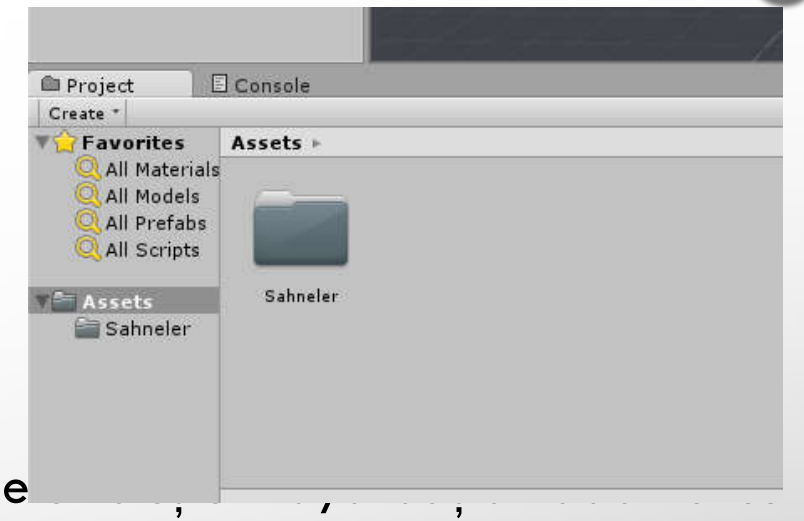
- File / New Project
- Projemiz 2D / 3D olabilir. 3D seçilir
- Proje klasörü belirlenir ve isim verilir.

- Proje açıldığında yeni bir scene ile açılır. Yeni nesneler oluşturmaya başlamadan önce bu scene kaydedilir.

- Genel olarak proje klasörü altında 'Scenes' yada benzeri şekilde bir klasör altına sahneler toplanabilir. Yapılan proje içinde birden fazla sahne olabilir ve bunlar arasında geçişler olabilir.

ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)



Yeni bir proje oluşturalım.

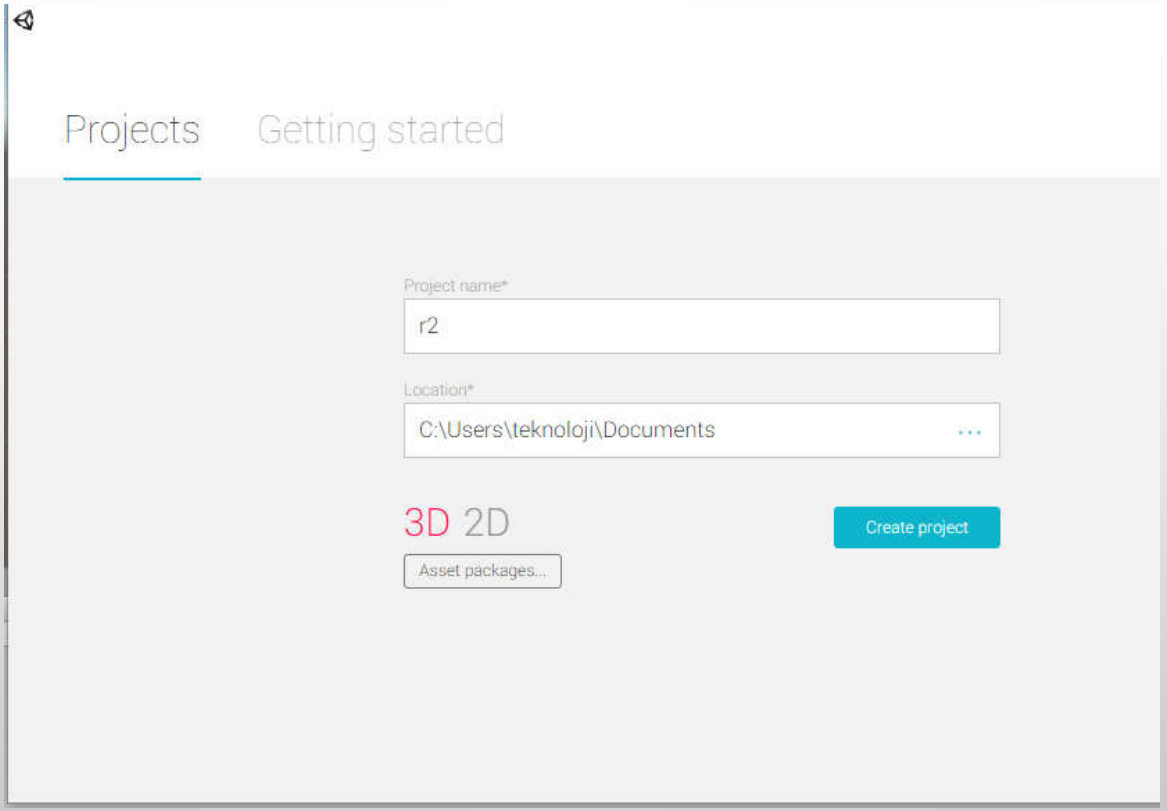
- File / New Project
- Projemiz 2D / 3D olabilir. 3D seçilir
- Proje klasörü belirlenir ve isim verilir.

- Proje açıldığında yeni bir scene ile açılır. Yeni nesne , , , bu scene kaydedilir. (File/Save Scene)

- Genel olarak proje klasörü altında *Assests* klasörü içine '***Scenes***' yada benzeri şekilde bir klasör altına sahneler toplanabilir. Yapılan proje içinde birden fazla sahne olabilir ve bunlar arasında geçişler olabilir.

ROLL-A-BALL örneği

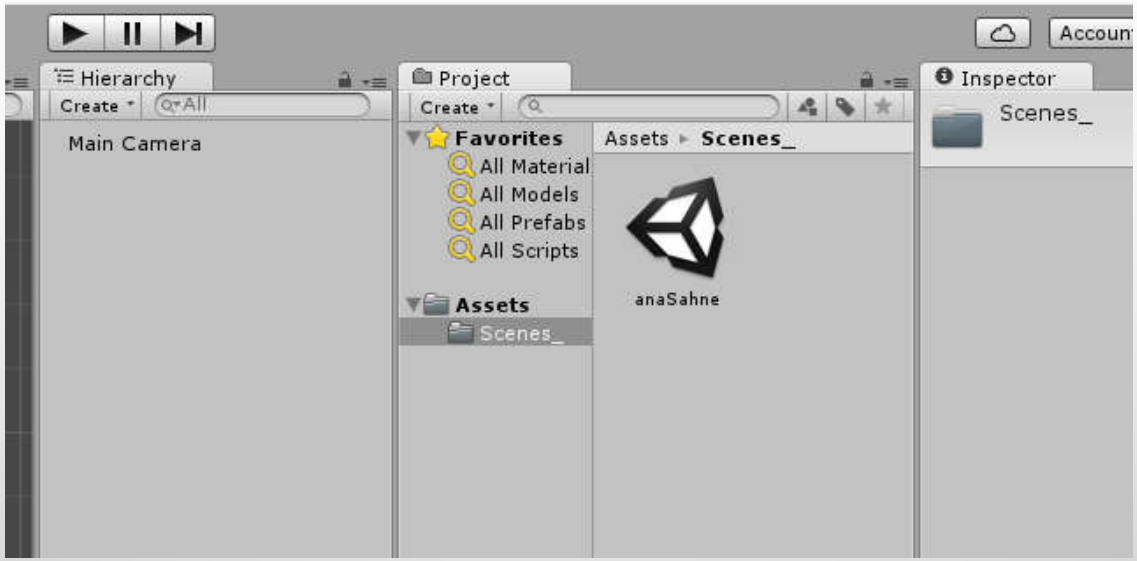
2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)



Yeni 3d Proje oluşturma

ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)



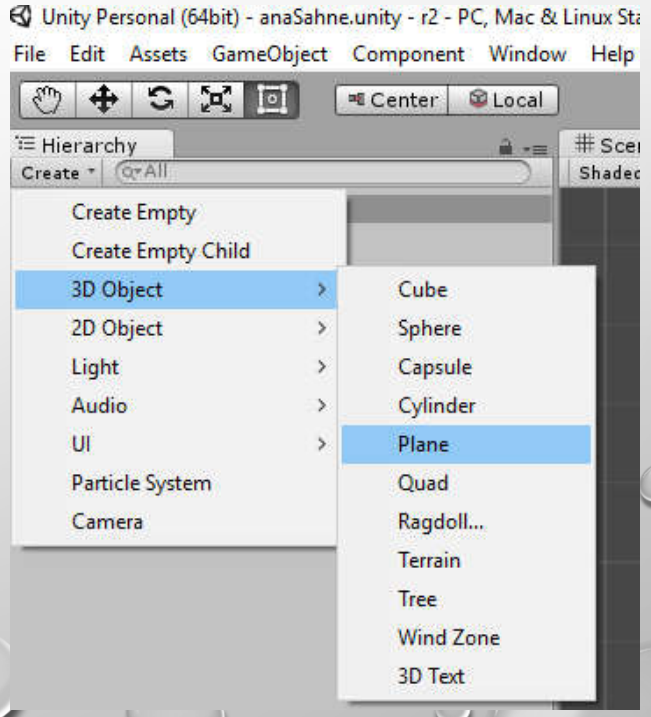
Sahne kaydetme

ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

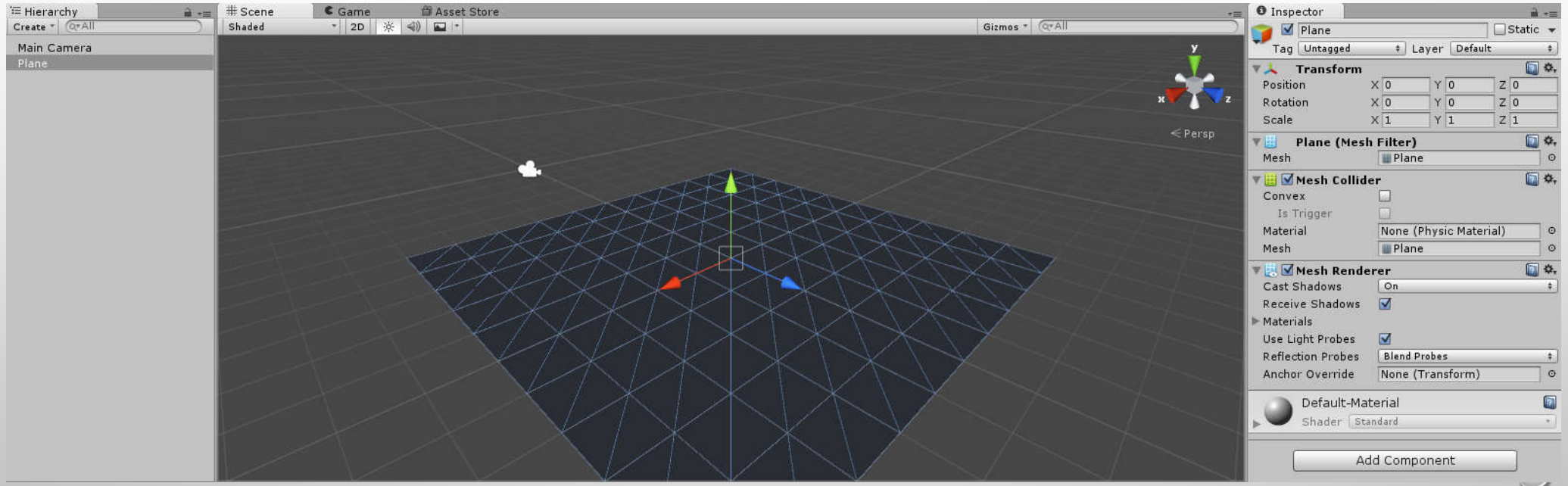
- Oyun sahnesi hazırlandıktan sonra oyun tahtasını ekleyelim. Ekleyeceğimiz obje 3D/Plane.

- Hierarchy menüsünden 3D/Plane
- GameObject menüsünden 3D/Plane



ROLL-A-BALL örneği

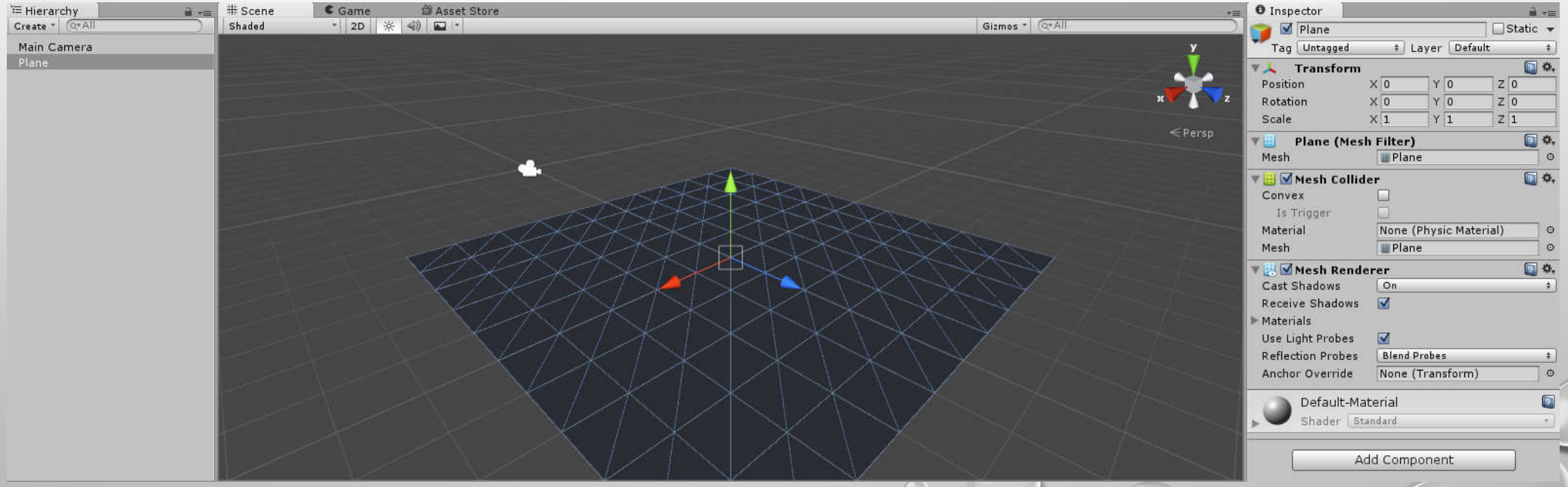
2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)



ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

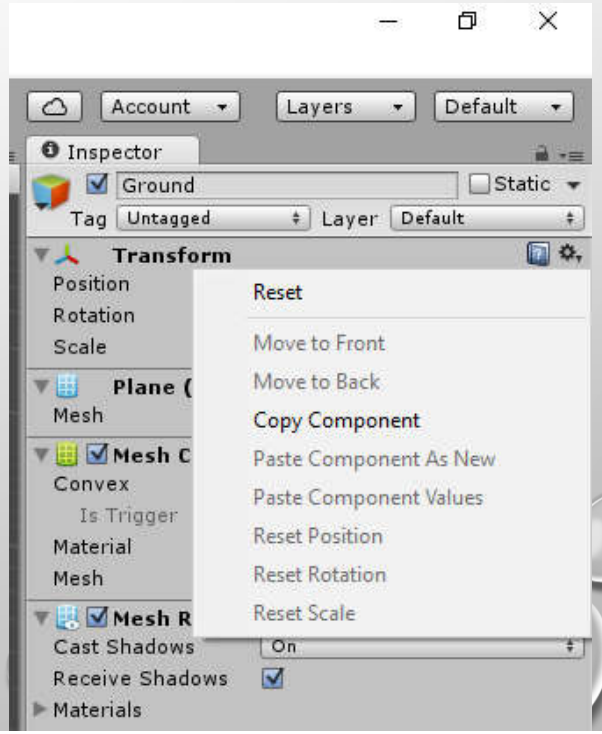
- Eklenen bileşenlere isimleri Inspector penceresindeki ‘Static’ kısmından verilir.
- Ayrıca hiyerarşi menüsünden nesne üzerinde aralık çift tıklanarak yada F2 tuşu ile.



ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

- Ground nesnesini merkez noktaya çekeriz. Bu koordinat değerleri World koordinat değerleridir.
- Inspector penceresindeki Gear simgesi tıklanır.
- Reset seçeneği ile nesne World koordinat için sıfır koordinatına çekilmiş olur.

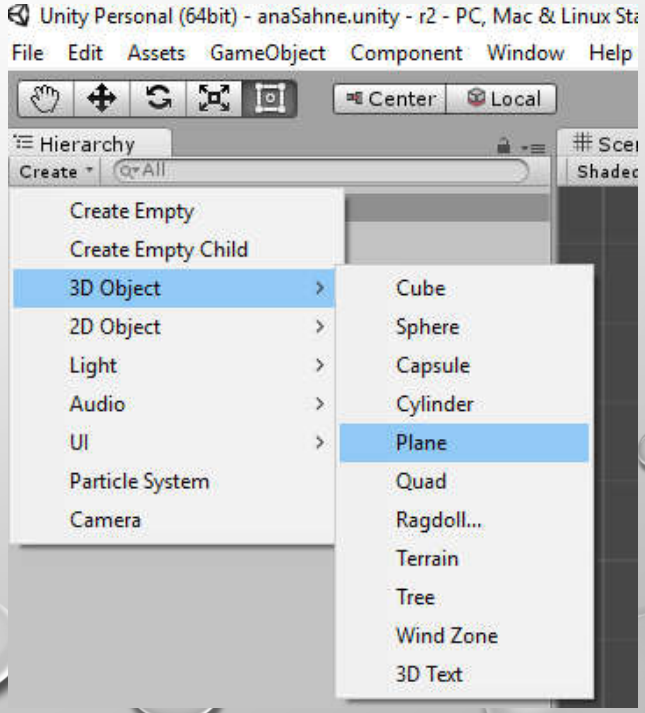


ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

- Oyuncu objesini oluşturmak için sahneye yeni bir 3D nesne ekleyelim. Eklenecek obje Küre(Sphere). Oyun içinde hareket edecek olan topumuz bu obje olacaktır.
- Obje eklenip koordinatları RESETlenir.

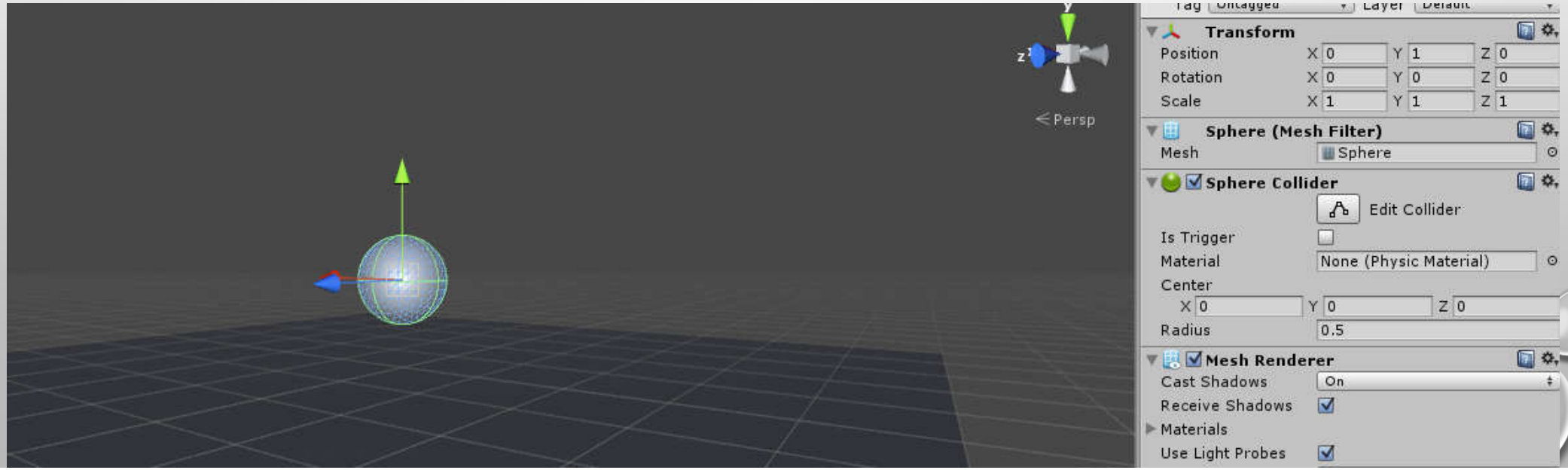
- Hierarchy menüsünden 3D/ Sphere
- GameObject menüsünden 3D/Sphere



ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

- Topu ekledikten sonra sahne üzerinde bir objeyi tam ekranda göstermek için obje seçilir ve 'F' (Frame) tuşuna basılır. (Yada Edit / Frame)
- Eklenen kürenin Y koordinatı *Inspector /Transform / Y* kısmından '1' yapılır. Eklenen her obje 1 birim şeklidedir.



ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

- Sahnemize ‘Skybox’ eklyelim. Skybox unity stock içinde mevcut arkaplan için kullanılabilir.

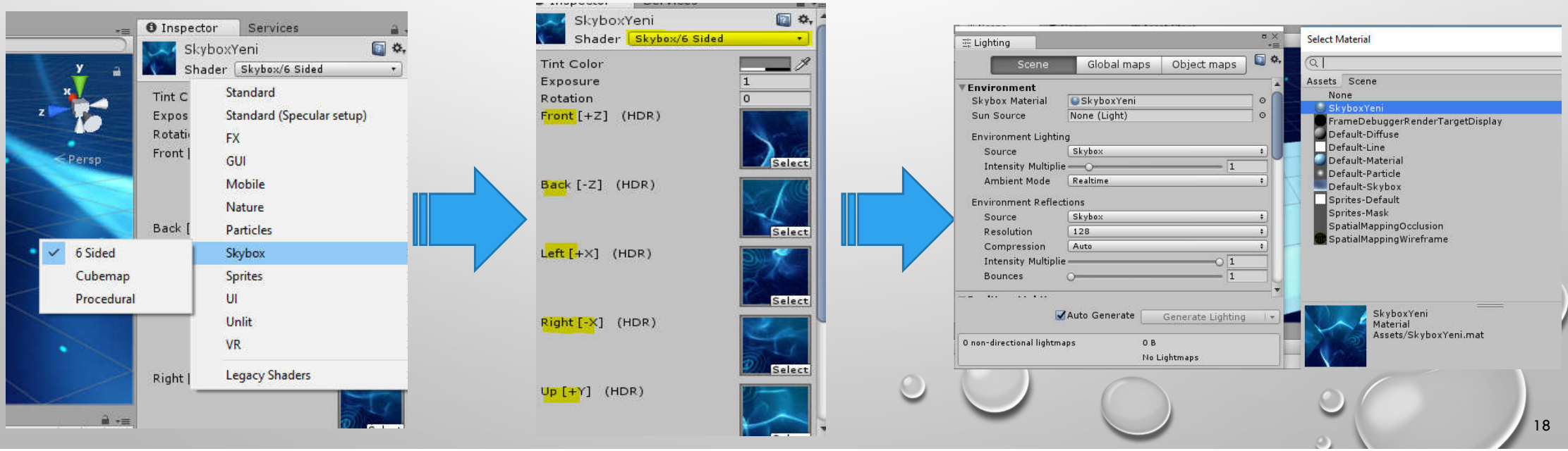
  Skybox eklemek için;

- Window / Lighting
- Scene kulakçığında Ortam aydınlatması (Environment lighting)
- Skybox kısmından mevcut materyallerden biri seçilip uygulanabilir. Default-Skybox seçilir.

ROLL-A-BALL örneği

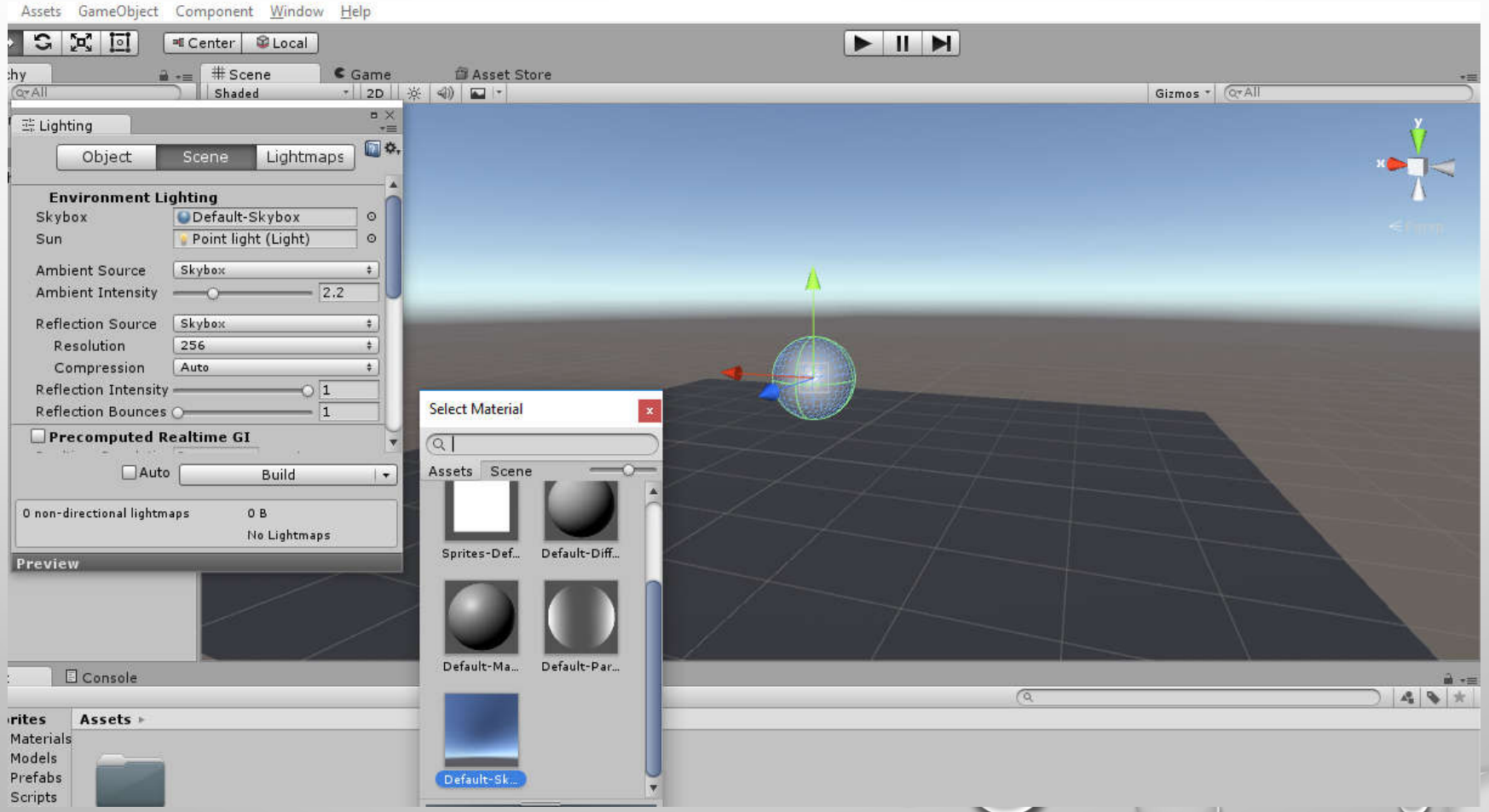
2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

- **Skybox 6 sided bir box nesnesidir.**
- Yeni bir materyal ekleyip Shader ayarını Skybox 6 sided ayarlayıp. Uygun resimleri hazırlayıp. Daha sonra Window/Lighting kısmından bu yeni materyali atayıp sahneye skybox verebilirsiniz. Ayrıca yine buradan FOG (sis) eklenebilir.



ROLL-A-BALL örneği

2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)



ROLL-A-BALL örneği

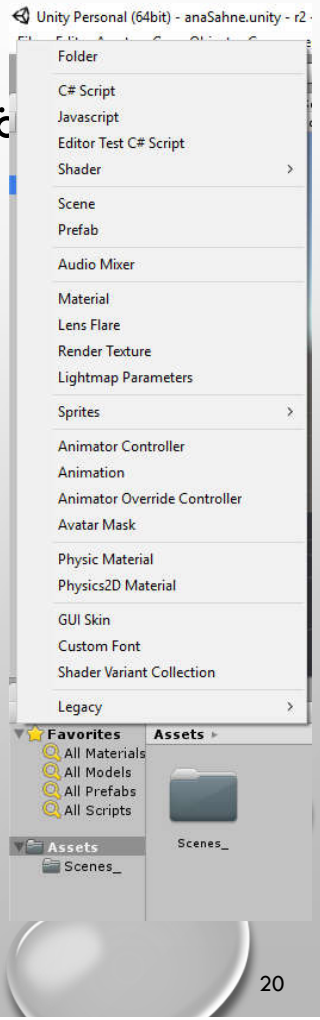
2. ENVIRONMENT VE OYUNCU AYARLARI (DEVAM)

✓ MATERYAL EKLEME

- Sahneye standart materyal eklemek için 'Materyal' isimli bir klasör oluşturulur. Bu klasö materyallerimizi tutacak. Ardından materyal oluşturulur. İsim verilir.

- *Project panel / Create / Folder*
- *Project panel / Create / Material*

- Inspector penceresinde 'Main Maps' kısmında 'Albedo' seçeneği ile Materyal rengi belirlenir. Preview ekranından sonuç görülebilir.
- Materyali ground'a(oyun sahamız) eklemek için oluşturulan materyal ground nesnesi üzerine sürüklenip bırakılır.



ROLL-A-BALL örneği

2. OYUNCUYU HAREKET ETTİRME

- Oyuncuyu hareket ettirmek için Fizik kurallarını uygulamalıyız. Oyun esnasında toplanacak nesnelere dokununca bu nesneleri toplayarak puan kazanacağız
- Fizik kurallarını uygulamak için;
- *Component / Physics / Rigidbody.*
- *Inspector panelinden 'Add Component' kısmından Physics / Rigidbody*

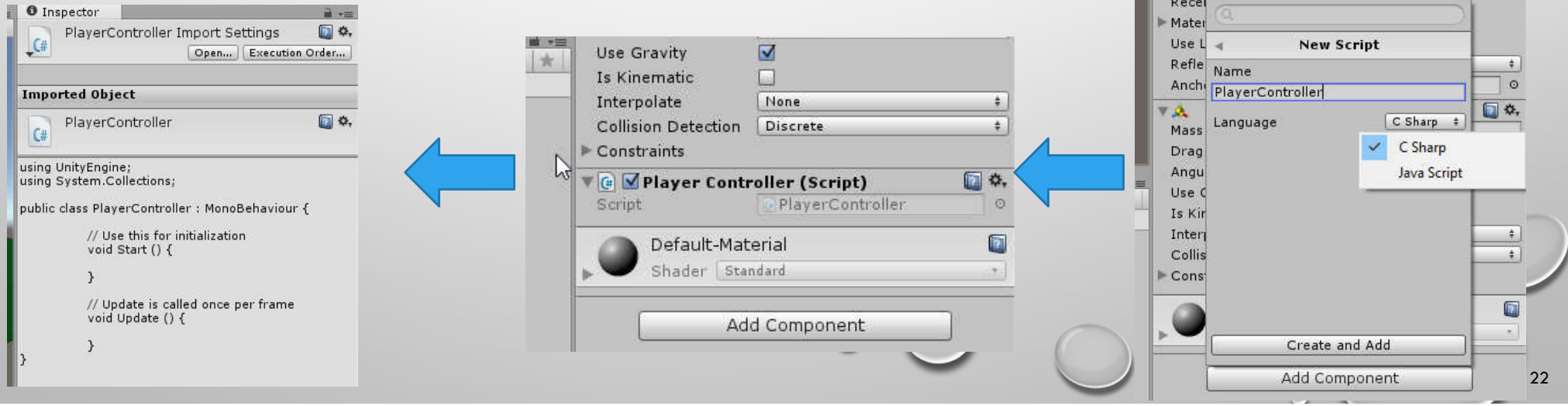
- Objeyi hareket ettirmek için C# scripti yazacağız. Script yazmadan önce 'Scripts' isimli bir klasör oluşturup buraya yeni bir script ekleyeceğiz

ROLL-A-BALL örneği

2. OYUNCUYU HAREKET ETTİRME (DEVAM)

- Script oluşturmak için Create menüsü , Project paneldeki Create seçeneği kullanılabilir.
- Uygulamamızda ise nesne seçilip Inspector'daki 'Add Component' seçeneğinden yeni C# scripti ekleyeceğiz.

- *Add Component / New Script / Add*
- *Script'i Assests klasöründen Scripts klasörüne taşırız*



ROLL-A-BALL örneği

2. OYUNCUYU HAREKET ETTİRME

- Script'i edit yaparak ekleyeceğimiz kod ile klavyeden gelecek inputlara göre topumuzu hareket ettireceğiz.
- Player(oyun topu) objesi seçili iken Gear ikonundan

'Edit Script' ile ilgili editör açılır ve kod buraya yazılır.

(Mono yada Visual Studio)

- Visual Studio açıldığında sample kod ile açılır. Sample kodu silip kodumuzu yazacağız.

ROLL-A-BALL örneği

2. OYUNCUYU HAREKET ETTİRME (DEVAM)

- Yazacağımız kod nasıl olacak? Kullanıcıdan gelecek olan bilgiyi almak istiyoruz. Her frame hareketinde kullanıcı bilgisini kontrol etmek istediğimiz için '*Update*' '*Fixed Update*' kullanılacak.
- *Update:* frame render edilmeden önce çalıştırılır. Buraya oyun kodu yazılacak.
- *FixedUpdate*: Fiziksel bir hesaplamadan önce çalıştırılır. Fizik kodumuz buraya yazılacak.

ROLL-A-BALL örneği

2. OYUNCUYU HAREKET ETTİRME (DEVAM)

- Input sınıfı kullanılarak giriş alınacak.
- (https://docs.unity3d.com/ScriptReference/Input.html)
- Fizik kuralları için *Rigidbody* kullanılacak
- https://docs.unity3d.com/ScriptReference/Rigidbody.html
- *AddForce()* fonksiyonu ile objeye güç uygulanır. *Vector3* x,y,z için değeri.
- Rigidbody için referans verilecek. Bu referans *Start()* fonksiyonu ile atanacak.
- Component ile rigidbody arasında bağıntı *GetComponent* metodu ile olacak.
- Hız değerini ayarlamak için ise public bir değişken katsayı olarak dışardan girilebilecek.

ROLL-A-BALL örneği

2. OYUNCUYU HAREKET ETTİRME (DEVAM)

```
using UnityEngine;
using System.Collections;

[RequireComponent(typeof(Rigidbody))]
public class PlayerController : MonoBehaviour {
    private Rigidbody rb;
    [Tooltip("hız değerini belirler")]
    public float hiz;
    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }
    void FixedUpdate()
    {
        //Klavye bilgisini alır
        float moveHor = Input.GetAxis("Horizontal");
        float movVer = Input.GetAxis("Vertical");
        //Vector3 movement = new Vector3(x, y, z);
        // x: yatay
        // y: 0.0f yukarı hareket yok
        // z: dikey
        Vector3 movement = new Vector3(moveHor,0.0f, movVer);
        rb.AddForce(movement*hiz);
```

ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI

✓ KAMERAYI HAREKET ETTİRME

- Main kamera seçildikten sonra inspector penceresinden ana kamera ayarları yapılır.

  Position : X: 0 Y: 10 Z:-10

  Rotation-Tilt : X: 45 derece

- Ana kamera Player (oyuncu) objesinin child'ı olacak şekilde sürükle bırak ile Player objesine bağlanır. Player hareket ettikçe kamera bakışı player'a göre olacaktır. Bunu görmek için layout 2-3 moduna alınır player hareket ettirilir.

- Fakat kamera objeye bağlı olduğunda topun rotate hareketi olduğu için istenmeyen bir durum oluşur. Bunun üstesinden gelmek için kamerayı player'a script ile bağlayacağız.

ROLL-A-BALL örneği

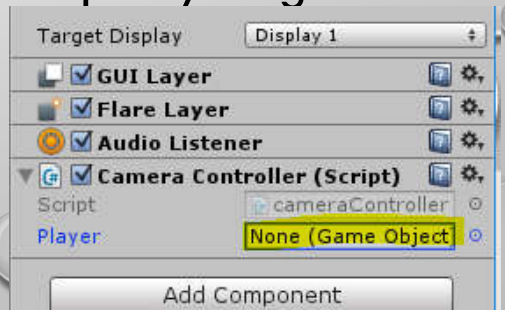
3. KAMERA ve OYUN ALANI

- Ana kamerayı child poziyonundan çıkarıp eski haline getirelim.
- Kamera seçili iken *Add Component* ile yeni bir C# scripti ekleyelim. Kamera için bir script yazılıp bu script player objesi ile eşleştirilecek.
- Script sonrası Player objesini kameranın Script kısmındaki Player değişkenine refere edip uygulamayı çalıştıracağız.
- A(x,y,z)=A(x',y',z')+OFFSET

ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI

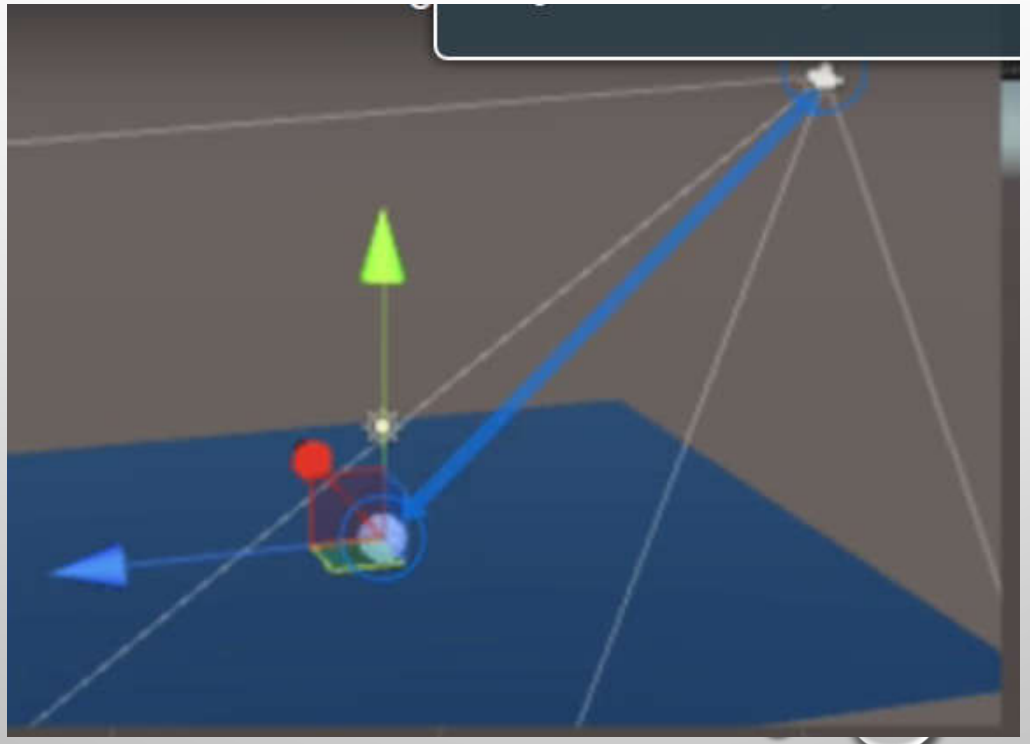
- Script nasıl çalışacak?
- En başta uygulama başladığında kamera pozisyonu ile player pozisyonu arasındaki fark(offset) hesaplanacak. Bu fark player objesinin her hareketinde tekrar tekrar hesaplanacak.
- Top(player) hareket ettikçe top ile kamera arasında bu fark her zaman korunacak. Böylece kamera topu takip etmiş olacak.
- *LateUpdate*: Bütün sahne render edildikten sonra çağrılır böylece topun yeri garanti edilmiş olur.



ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI

- Offset değeri gösterimi. Bu fark sürekli korunacaktır.



ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI

- Script çalıştırılmadan önce koddaki public tanımlı olan *GameObject* türündeki *player* değişkeni sahnedeki player(topumuz) nesnesine refere edilmeli. Bunun için Hiyerarşi menüsünden player nesnesi scriptteki public kutuya sürüklenip bırakılmalı

ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI

-Kamera Kontrol C# Kodu:

```
using UnityEngine;
using System.Collections;

public class CamControl : MonoBehaviour {

    private Vector3 offset;
    public GameObject player;
// Use this for initialization
void Start () {
        offset = transform.position - player.transform.position;
}

// LateUpdate is called after all
void   LateUpdate () {
        transform.position = player.transform.position + offset;
}
}
```

ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI

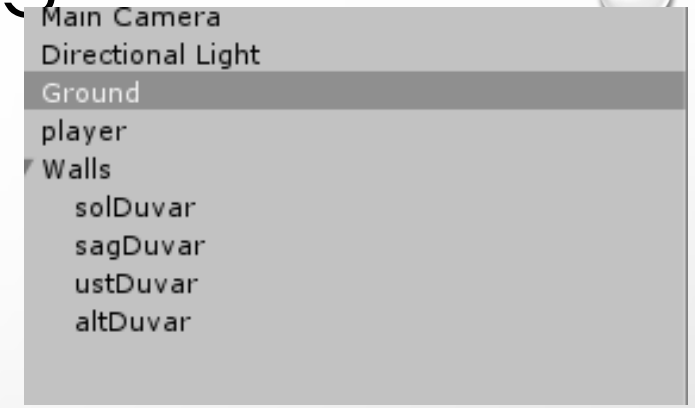
✓ OYUN ALANI AYARLARI

- Bu kısımda nesnelerin düşmemesi için duvarlar oluşturulacak. Ayrıca oyun esnasında kendilerine dokunulduğunda toplanabilir nesneler oluşturulacak.
- Duvarları boş bir obje başlığı hiyerarşisinde tutacağız. *Walls* isimli bir boş oyun nesnesi oluşturup altına duvarları hiyerarşik olarak ekleyeceğiz.

  -Hiyerarşi panelinden Create / Empty
- Oluşturulan Walls nesnesini orjine Resetlenir.

ROLL-A-BALL örneği

3. KAMERA ve OYUN ALANI



✓ OYUN ALANI AYARLARI

- Duvalar için Cube 3d objesi kullanılacak.

  - Hiyerarşi panelinden Create / 3D / Cube

- Oluşturulan nesne resetlenip nesneye isim verilir. Yükseklik ve genişlik değerleri verilir ve oyun tahtasının ilgili kenarına taşınır.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma

b) Topları toplama

c) Skoru gösterme

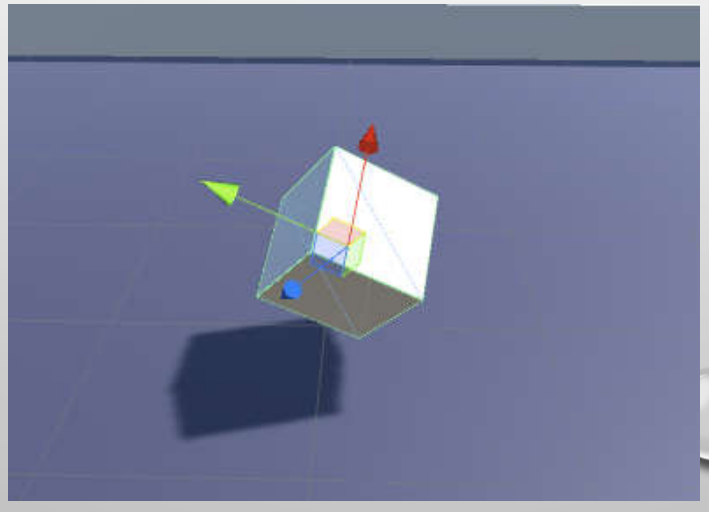
d) Oyunu derleyip çalıştırma

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma

- Toplanabilir objeler için yine 3d/Cube nesnesi eklenir ve resetlenir. Player objesi geçici olarak kapatılır.
- Eklenecek Cube nesnesi X,Y,Z değerleri ayarlanarak şekildeki hale getirilir.
- Ayrıca tilt değerleri değiştirilir. 3 açı değerine de 45 derece değeri girilir.



ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma

- Oluşturulan bu objenin daha güzel görünmesi için bu nesne döndürülebilir. Böylece oyun içinde bir hareketlilik sağlanmış olur.
- Rotasyon işlemi için script kullanılır. Elmas (pickup obje) seçili iken
  - *Inspector Panel / Add Component / New Script*
- C# scriptine isim verilir Vstudio açılır.
- Örnek kod içinde Update kısmı kullanılarak rotasyon işlemi gerçekleştirilecektir.
- Yeni dosya Script klasörüne taşınır.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma

- Kod ile yapılacak olan en başta verilmiş olan tilt değerini (45derece) her bir eksen için kod ile değiştirerek kübün dönmesini sağlamak.
- Kod içinde Update kısmı kullanılır. *transform* sınıfı ile rotasyon gerçekleşirilir. (https://docs.unity3d.com/ScriptReference/Transform.html)
- Transform içinde rotasyon için 2 fonksiyon kullanılır. *Rotate* ve *Translate*. Burada açı değerleri verilerek uygulama yapılacağı için Rotate kullanılmaktadır. (https://docs.unity3d.com/ScriptReference/Transform.Rotate.html)
- Rotate fonksiyonunda Vector3 yada X,Y,Z değerleri verilerek rotasyon yapılır. Burada Vector3 kullanılmaktadır.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma: C# kodu

```
using UnityEngine;
using System.Collections;

public class rotator : MonoBehaviour {

// Update is called once per frame
void Update () {
        transform.Rotate(new Vector3(15, 30, 45)*Time.deltaTime);
//Time.deltaTime : frame bağımsız hale getirmek için
//örn: bir framede 10mt gitmek yerine saniyede 10mt git demek gibi.frame bağımsız
}
}
```

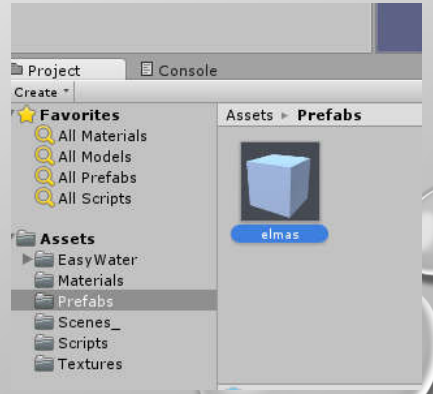
Test edip nesnenin rotasyonunu görebiliriz.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma

- Bu noktadan sonra toplanacak objemizi artık *Prefabs* olarak tanımlayacağız. Prefab'lar objelerin şablonları yada izleri gibidir.
- Prefabs olarak tanımlanan objeler diğer sahnelerde de eklenebilir nesneler haline gelir.
- Bunun için 'Prefabs' isimli bir klasör oluşturulur. Obje yeni oluşturulan Prefabs klasörüne taşınır.



ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

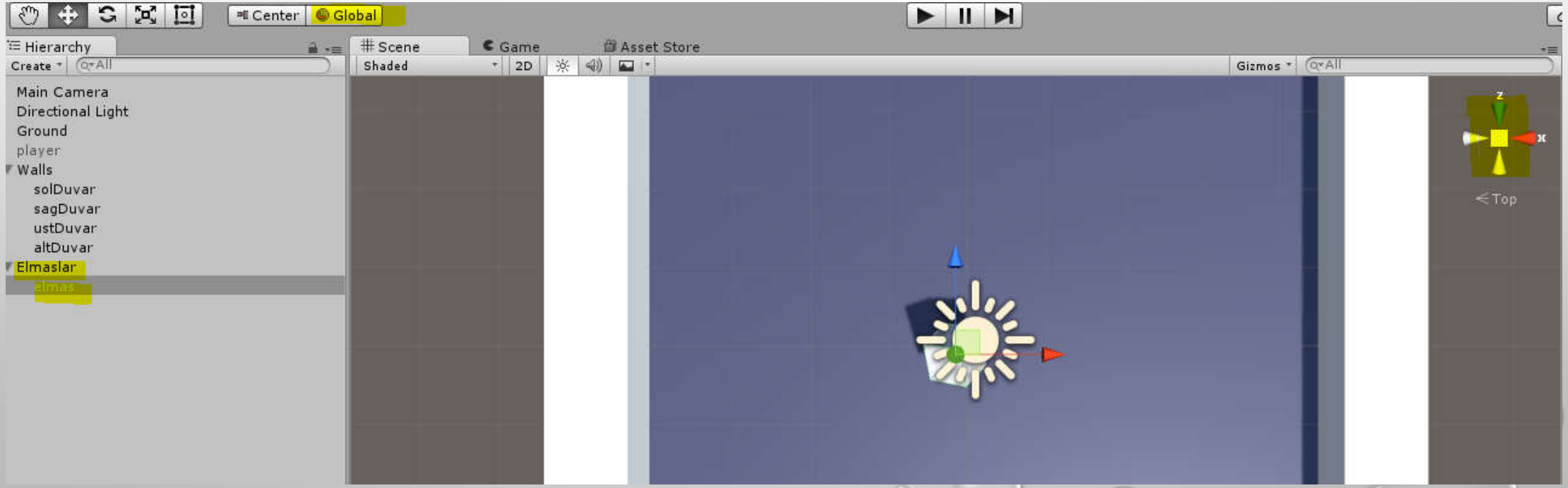
a) Toplanabilir nesneler oluşturma

- Toplanacak elmas nesneleri *Pickups* isimli bir başlık altında toplanır. Prefabstan çekilecek elmas objeleri oyun alanı üzerine belli aralıklarla yerleştirilir.
- Bu yerleştirme yapılmadan önce sahneye Y ekseninden bakılarak, koordinat sistemi *Local* seçeneğinden *Global'a* çekilir. Böylece obje 45lik yerel koordinat yerine global koordinata göre hareket eder. X ve Z eksenlerinde hareket ettirilecektir.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma



ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

a) Toplanabilir nesneler oluşturma

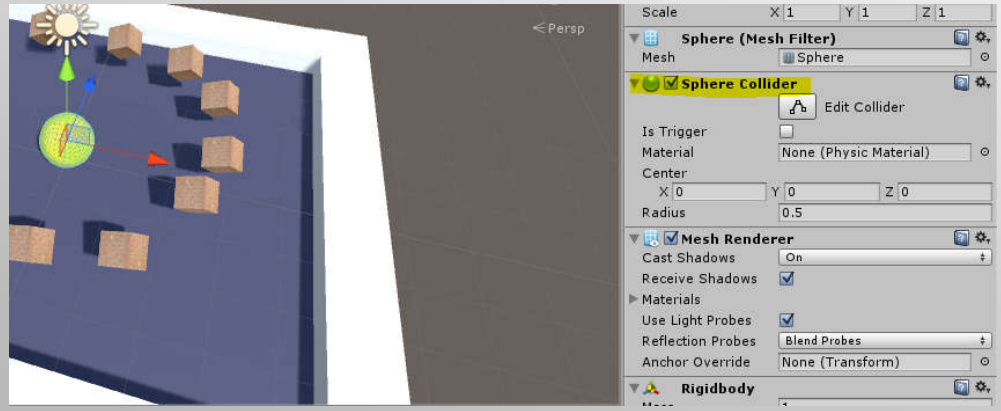
- Elmaslara yeni bir materyal ile renk yada texture verilir.
- Texture için belli bir resim dosyası Texture isimli klasöre taşınır.
- Texture klasöründen bileşene sürükle bırak ile Texture atanır. Ardından Inspector penceresinden Apply ile tüm prefab'e uygulanır.
- İkinci yol ise, texture direk olarak Prefab klasöründeki prefab nesnesine sürükle bırak ile atanır.

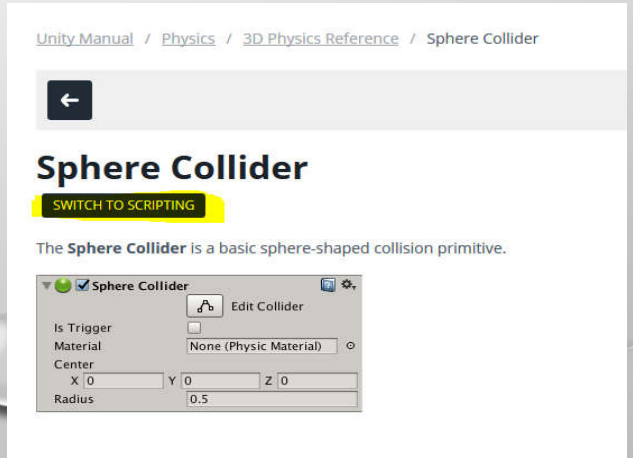
ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

b) Elmasları Toplama (Collision ve Trigger)

- Elmasları toplamak için Collider sınıfından yararlanılır. Sahnedeki nesnelerin Collider şekilleri bulunmaktadır. Collision olacak obje kontrol edilerek istenen obje ile temasa geçildiğinde tetikleme gerçekleştiriliz.
- *Player* objesinin C# kodu içine script yazılır.
- Ya da Inspector penceresinde seçili objenin Collider paneli kullanılır.





ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

b) Elmasları Toplama

- Sphere Collider için '**OnTriggerEnter**' metodu kullanılır. Eğer nesne başka bir Collider ile çarpışacak olursa nesne yok edilecek böylece Elmas toplanmış olacaktır.
- Collider sınıfı ile ilgili referans.

(https://docs.unity3d.com/ScriptReference/Collider.html)

- Collider çeşitleri
  - BoxCollider
  - Sphere Collider
  - Capsule Collider
  - Mesh Collider

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME (OnCollisionEnter vs OnTriggerEnter)

b.1) Elmasları Toplama: Player script'i altına bu metod eklenir.Bu eventin çalışması fizik kurallarına göre değil koda göre tetikleme esasına göredir. Yani onTriggerEnter olayı için kod yazılır. Elmas nesnesinin isTrigger seçeneği açık olmalıdır.

```
void OnTriggerEnter(Collider other)
    {
        //Destroy(other.gameObject);
        if (other.gameObject.CompareTag("elmas"))
        {
            other.gameObject.SetActive(false);
        }
    }
```

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME (OnCollisionEnter vs OnTriggerEnter)

b.2) Eğer fizik kurallarına göre bir çarpışmadan bahsedecek olursak ve isTrigger kapalı ise o zaman onCollisionEnter event'i kullanılabilir. Ayrı ayrı çalıştırıp fark görülebilir. Destroy ile nesne yok edilir. Setactive sadece gizler!

```
private void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("elmas"))
            Destroy(collision.gameObject);
        ////yada Find ile bulup DESTROY ile yok etmek!
        //duvar = GameObject.Find("ustDuvar");
        //Destroy(duvar);
        Debug.Log(collision.contacts[0].point);
        Debug.Log(gameObject.name + " nesnesi," + collision.gameObject.name + " nesnesine
çarptı");

    }
```

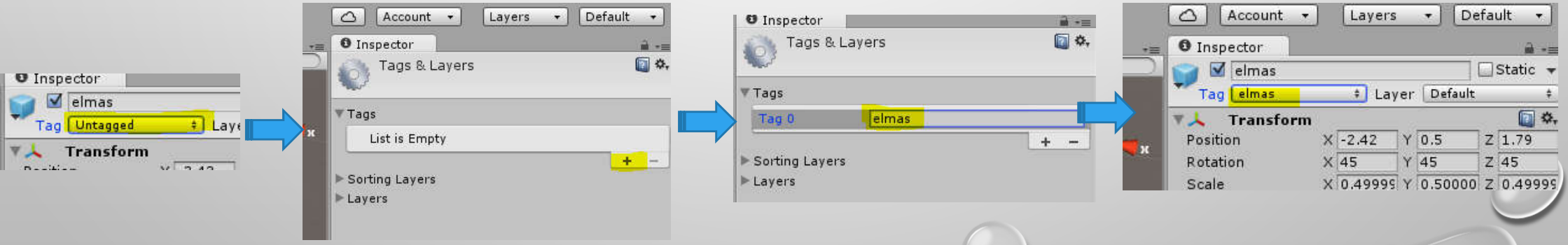
ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

b.3) Elmasları Toplama:

Koddaki «elmas» isimli tag için yeni bir etiket verilmeli. Bunun için Prefab içindeki elmas objesine Inspector penceresinden yeni bir Tag eklenir. Elmas objesi seçili iken;

- Inspector paneliden / Tags / Add Tag

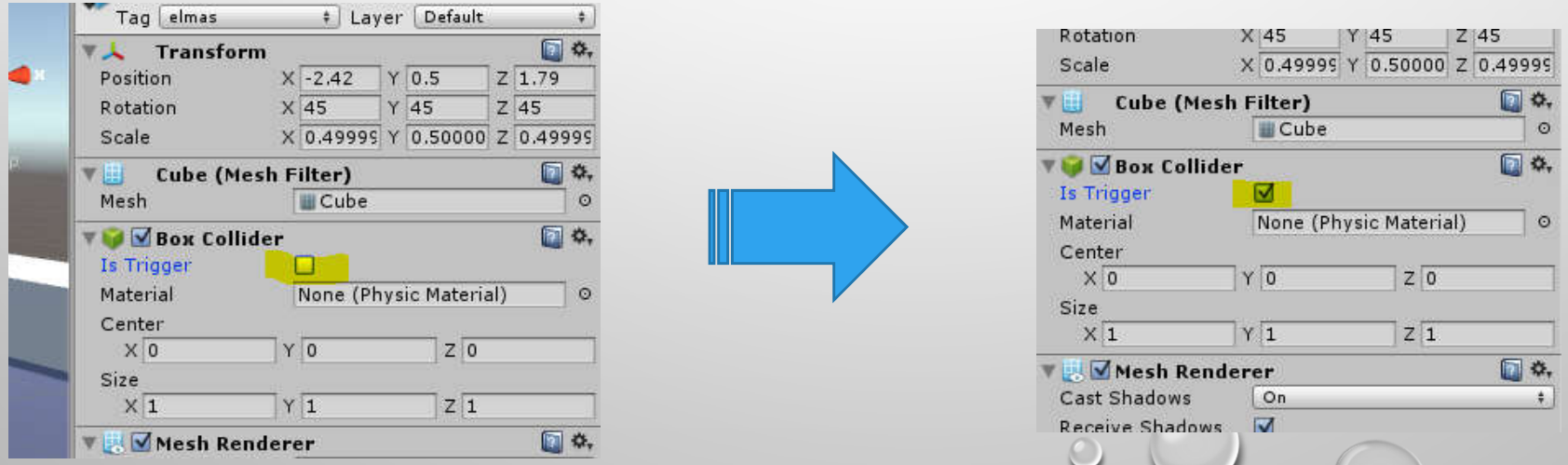


ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

b.4) Elmasları Toplama:

Çarpışma esnasında trigger'ın devreye girmesi için. Prefab/elmas objesi seçilip Inspector penceresinden 'Is Trigger' aktif hale getirilir. Uygulama çalıştırılır.OnTriggerEnter eventinin çalışması için. Eğer ***isTrigger*** seçilmemiş ise ***OnCollisionEnter*** kullanılmalıdır.



ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

c) Skoru hesaplama & yazdırma

Skor için C# scripti içine Sayac isimli private bir değişken tanımlanır, Start() içinde bu değişken sıfırlanır. Bu değişkenin değeri *OnTriggerEnter* içinde çarpışma anında bir arttırılır.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

c) Skoru hesaplama & yazdırma

C# Scripti;

```
private int sayac;
//Start içinde sıfırlanır
 void OnTriggerEnter(Collider other)
    {
        //Destroy(other.gameObject);
        if (other.gameObject.CompareTag("elmas"))
        {
            other.gameObject.SetActive(false);
            sayac++;
        }
    }
```

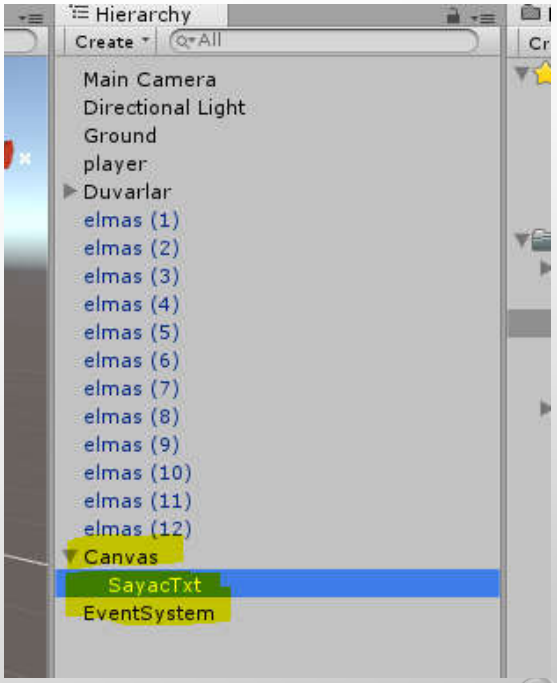
ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

c) Text UI ile Skor ve Mesaj Yazdırma (UI Text)

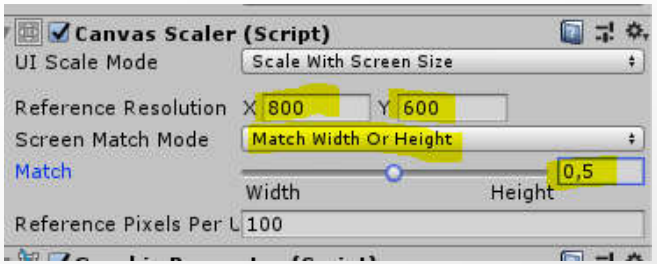
Elmaslar toplanırken hesaplanan sayaç değeri skor olarak UIText elemanı kullanılarak ekranda gösterilir. Tüm elmaslar toplandığında da oyun sonu mesajı yazdırılır.

Bir metni göstermek için *Create/UI Text* elemanı oluşturulur. ***SayacTxt*** ismi verilen text bileşeni oluşturulduğunda bu bileşen için otomatik olarak *Canvas ve EventSystem* de oluşturulur.



ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME



CANVAS UI SCALE MODE

Canvas sistemin çözünürlüğe göre ayarlamak gereklidir. Farklı çözünürlükte oyun alanı benzer görülmeli genelde 2D oyunlar için yapılan bir ayardır. Canvas'ı seçip UI Scale mode ile Oyunun GAME kulakçığındaki çözünürlüğe ayarlayıp, Match kısmınıda ortada dengeli 0,5 değeri verip Width ve Height için dengeli eşleştirme değeri verilir.

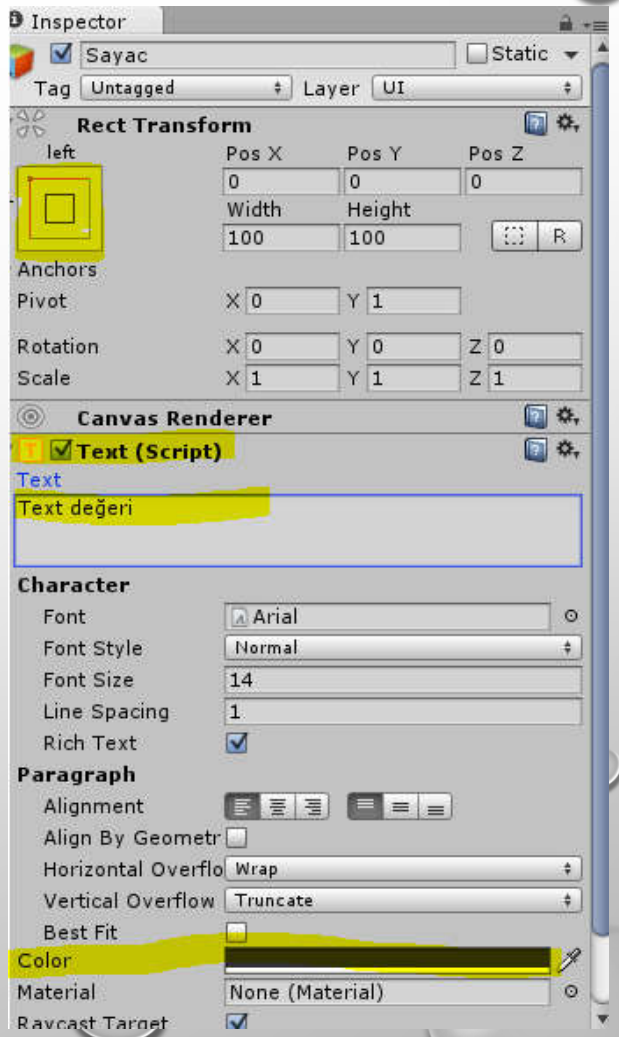
ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

c) Skor ve mesaj Yazdırma (UI Text element)

Oluşturulan text sol alt tarafta görülecektir. Text'e renk değeri verilir. Anchor özelliği ekranda istenen noktaya text konumlandırılır.

- Color ile renk verilir.
- Anchor ile sol üst köseye text sabitlenir.
- Marjin ayarı için PosX(10) ve PosY(-10) ayarlanır.



ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

c) Skor ve mesaj Yazdırma (UI Text element)

sayacText ile ***Sayac*** değişkeninin birbirine bağlanması için kod içinde yeni bir NameSpace eklenmesi gereklidir.

- Text UI nesnesi için ***UnityEngine.UI*** namespace'i koda dahil edilir.
- Public olarak Text tipinde ***sayacText*** isimli bir değişken tanımlıyoruz.
- Start() ile bu değişkene sayaç değeri aktarılır.
- Tetiklemenin olduğu her olayda da sayac değişkeninin değeri bir arttırılırken bu değer ***sayacText*** değişkenine atanır.
- Eğer toplanan elmas sayısı toplam elmas sayısına eşit ise «Kazandınız» şeklinde yeni bir mesaj yazdırılıp oyun sonlandırılır.

ROLL-A-BALL örneği

4. OYUN SKORU ve DERLEME

c) Skor ve mesaj Yazdırma (UI Text element)

C# kodu;

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;

public class PlayerController : MonoBehaviour {

    private Rigidbody rb;
   public float hiz;
    private int sayac;
    public Text sayacText;
    public Text kazandinText;
// Use this for initialization
void Start () {
        sayac = 0;
        kazandinText.text = "";
        sayacText.text =  sayac.ToString();
        rb = GetComponent<Rigidbody>();

}
```

```
void FixedUpdate()
    {
        float mHor = Input.GetAxis("Horizontal");
        float mVer = Input.GetAxis("Vertical");
        Vector3 hareket = new Vector3(mHor, 0.0f, mVer);
        rb.AddForce(hareket*hiz);
    }

void OnTriggerEnter(Collider other)
    {
        //Destroy(other.gameObject);
        if (other.gameObject.CompareTag("elmas"))
        {
            other.gameObject.SetActive(false);
            sayac++;

            sayacText.text = sayac.ToString();
            if (sayac == 12)
                kazandinText.text = "KAZANDINIZ!!!!";
        }

    }
```

ROLL-A-BALL örneği

4. OYUN SEVİYE (Scene) DEĞİŞTİRME

d) Sahne (scene) değiştirme

- Oyun skoru yazdırıldıktan sonra, tüm normal elmaslar toplandıktan sonra, **özel elmas** alınınca bir süre bekleyip(3sn) yeni sahne(level) için geçiş yaptırılabilir.

- Bunun için yeni bir 'cikisSahnesi' eklenir.Çıkış sahnesi içine bir UI text ekleyip '**GameOver**' yazısı yazdırılır.
- Sahne işlemleri Unity.SceneManager ile yönetilir.
- Özel elmas tag'ı farklı olan bir elmas olacaktır.

ROLL-A-BALL örneği

4. OYUN SEVİYE (Scene) DEĞİŞTİRME

d) Sahne (scene) değiştirme

- Sahne işlemleri Unity.SceneManager namespace ile yönetilir.Sahne BuildSettings ile eklenmelidir.

CreateScene	Create an empty new scene at runtime with the given name.
GetActiveScene	Gets the currently active scene.
GetSceneAt	Get the scene at index in the SceneManager's list of added scenes.
GetSceneByBuildIndex	Get a scene struct from a build index.
GetSceneByName	Searches through the scenes added to the SceneManager for a scene with the given name.
GetSceneByPath	Searches all scenes added to the SceneManager for a scene that has the given asset path.
LoadScene	Loads the scene by its name or index in Build Settings. [LoadScene adının veya Yapı Ayarlarında endeksi ile sahne yükler.]
LoadSceneAsync	Loads the scene asynchronously in the background. [LoadSceneAsync arka planda uyumsuz sahne yükler.]
MergeScenes	This will merge the source scene into the destinationScene.
MoveGameObjectToScene	Move a GameObject from its current scene to a new Scene.
SetActiveScene	Set the scene to be active.
UnloadSceneAsync	Destroys all GameObjects associated with the given scene and removes the scene from the SceneManager.

sceneCount	The total number of currently loaded scenes.
sceneCountInBuildSettings	Number of scenes in Build Settings.

ROLL-A-BALL örneği

4. SEVİYE (Scene) DEĞİŞTİRME

- Oyun skoru yazdırıldıktan sonra, tüm normal elmaslar toplandıktan sonra özel elmastan sonra, belli bir süre bekleyip(3sn) yeni sahne(level) için geçiş yaptırılabilir. (*LoadScene** ile)

  *LoadScene için 'cikisSahnesi'nin Build Settings'den sahne olarak eklenmesi gereklidir!

- Bekletme için ise bir rutin çalıştırılabilir. ***IEnumerator*** Interface arabirimi, ***Waitforseconds*** metodu ile belli bir süre bekleme sağlanabilir.
- Özel elması son alması için bir Text ile uyarı mesajı yazdırılabilir; Yada ***GUI.Box*** veya ***EditorUtility.DisplayDialog*** ile ekrana sabit mesaj yazılabilir.

ROLL-A-BALL örneği

4. OYUN SKORU ve SEVİYE (Scene) DEĞİŞTİRME

```
private void OnTriggerEnter(Collider other)
    {
        if(other.gameObject.CompareTag("elmas"))
        {
            other.gameObject.SetActive(false);
            sayac += 10;
            skor.text = "Skor:"+sayac.ToString();

        }
        else if (sayac == 70 && other.gameObject.CompareTag("sonElmas"))
        {
            other.gameObject.SetActive(false);
            sayac += 10;
            skor.text = "Skor:" + sayac.ToString();
            kazandin.text = "Tebrikler!!";
            StartCoroutine(Beklet());
        }
        else
        {
            UyariTxt.text = "Kırmızı en son alınacak!!";
            StartCoroutine(Uyari());
        }
    }
```

```
// EditorUtility.DisplayDialog("Kırmızı en son!!", "En son kırmızı", "Tamam!"
//// OnGUI();
   //private void OnGUI()
   //{
   //    GUI.Box(new Rect(600, 0, Screen.width / 4, Screen.height / 10),
"Kırmızı elması son al!!!");
   //}
```

```
IEnumerator Beklet()
{
    yield return new WaitForSeconds(3);
    SceneManager.LoadScene("cikisSahnesi");

}
IEnumerator Uyari()
{
    yield return new WaitForSeconds(2);
    UyariTxt.text = "";

}
```

ROLL-A-BALL örneği

4. OYUNU SONLANDIRMA ve DERLEME

Oyundan çıkış için; Update içine GetKey ile Esc tuşu için kontrol yaptırılıp Application.Quit ile çıkış yapılabilir. Sahnedeki MainCam için bu script yazılabilir.

```
private void Update()
    {
        //if (Input.GetKeyDown(KeyCode.Escape))
        //    Application.Quit();
        StartCoroutine(Beklet());

    }
    IEnumerator Beklet()
    {
        yield return new WaitForSeconds(2);
        Application.Quit();
    }
```

Jump

```
////if (Input.GetKeyDown(KeyCode.Space))
        //if (Input.GetButton("Jump"))
        //{
        //   // r.AddForce(transform.forward * hiz * Time.deltaTime);
        //    r.AddForce(new Vector3(0, 1000f, 0) * Time.deltaTime *
hiz);
        //}
```

ROLL-A-BALL örneği

4. OYUNU DERLEME

- Oyunu farklı platformlar için derlemek için File/BuildSettings-Run kısmı kullanılabilir.
- Uygulamanın çalışacağı platformu belirleyip ***Input*** sınıfında ***klavye*** yada ***gyro/acc*** kullanılabilir.
- Bunun için uygulamanın çalıştığı platform check edilip buna göre klavye yada gyro kullanılabilir. Çalışan platformu tespit etmek için ***Application*.platform** sınıfındaki ***RuntimePlatform*** enum kullanılır.

OSXEditor	In the Unity editor on macOS.
OSXPlayer	In the player on macOS.
WindowsPlayer	In the player on Windows.
WindowsEditor	In the Unity editor on Windows.
IPhonePlayer	In the player on the iPhone.
Android	In the player on Android devices.
LinuxPlayer	In the player on Linux.
LinuxEditor	In the Unity editor on Linux.

ROLL-A-BALL örneği

4. OYUNU DERLEME

- Oyunu farklı platformlar için derlemek için File/BuildSettings-Run kısmı kullanılabilir.
- Uygulamanın çalışacağı platformu belirleyip ***Input*** sınıfında ***klavye*** yada ***gyro/acc*** kullanılabilir.
- Bunun için uygulamanın çalıştığı platform check edilip buna göre klavye yada gyro kullanılabilir. Çalışan platformu tespit etmek için ***Application.platform*** sınıfındaki ***RuntimePlatform*** enum kullanılır. (Yada `if(SystemInfo.deviceType==DeviceType.Handheld) )`

```
if (Application.platform==RuntimePlatform.Android)
      {
          mVer = Input.acceleration.y;
          mHor = Input.acceleration.x;
      }
```

ROLL-A-BALL örneği

4. OYUNU DERLEME

```
    private void FixedUpdate()
        {
            float mVer=0, mHor=0;
            Vector3 hareket;
            //if(SystemInfo.deviceType==DeviceType.Handheld)
            if (Application.platform==RuntimePlatform.Android)
            {
                mVer = Input.acceleration.y;
                mHor = Input.acceleration.x;
            }
            else if(Application.platform==RuntimePlatform.WindowsPlayer)
            {
             mVer = Input.GetAxis("Vertical");
             mHor = Input.GetAxis("Horizontal");

            }
            else if(Application.platform==RuntimePlatform.WindowsEditor)
                {
                mVer = Input.GetAxis("Vertical");
                mHor = Input.GetAxis("Horizontal");
            }
            hareket = new Vector3(mHor, 0.0f, mVer);
            rb.AddForce(hareket * Hiz);


    }
```

ROLL-A-BALL örneği

4. OYUNU DERLEME

- Oyunu farklı platformlar için derlemek için File/BuildSettings-Run kısmı kullanılabilir.

- Player Settings kısmından Bundle ayarı yapılır
- Versiyon ayarı yapılır
- Uygulama title verilir
- Icon seçilir.

ROLL-A-BALL örneği

4. OYUNU DERLEME

-