Orijinal Kaynak: http://forum.unity3d.com/threads/unity-lesson-1-draft.103421/

Unity 3D İle Oyun Programlama

Bölüm 11: Düşmanın Hasar Vermesi

ÇEVİRİ: Süleyman Yasir KULA

http://yasirkula.com

Ferdinand Joseph Fernandez

Chief Technological Officer, Dreamlords Digital Inc.

Admin and Co-founder, Unity Philippines Users Group

September 2011

Biz zombilere hasar verebiliyoruz ama zombiler bize hasar veremiyor. Bu bölümde, zombilerin menzilimize girince bize hasar vermesini sağlayacağız.

Sağlığı Ekranda Göstermek

Hasar alıp almadığımızı kolayca anlamak için sağlığımızı ekranda göstermeliyiz.

Health scriptini güncelleyin:

```
01 using UnityEngine;
02 using  System.Collections;
03
04 public class  Health : MonoBehaviour
05 {
06       [SerializeField]
07       int  _maximumHealth = 100;
08
09       int  _currentHealth = 0;
10
11       override public  string  ToString()
12       {
13            return  _currentHealth + " / "  + _maximumHealth;
14       }
15
16       void  Start()
17       {
18            _currentHealth = _maximumHealth;
19       }
20
21       public  void  Damage(int  damageValue)
22       {
23            _currentHealth -= damageValue;
24
25            if  (_currentHealth <= 0)
26            {
27                 Animation a = GetComponentInChildren<Animation>();
28                 a.Stop();
29
30                 Destroy(GetComponent<PlayerMovement>());
31                 Destroy(GetComponent<PlayerAnimation>());
32
33                 Destroy(GetComponent<EnemyMovement>());
34                 Destroy(GetComponent<CharacterController>());
35
36                 Ragdoll r = GetComponent<Ragdoll>();
37                 if  (r != null)
38                 {
39                      r.OnDeath();
40                 }
41            }
42       }
43 }
44
```

Her scriptte otomatik olarak yer alan (ama scriptte gözükmeyen) ToString fonksiyonunu kendimiz yeniden yazıyoruz (override). Script bize mevcut canımızın maksimum canımıza olan oranını söylüyor; "94 / 100" gibi...

ToString'in döndürdüğü bu string'i ekranda göstermemiz lazım. Bunun için PlayerGui kodunu değiştirelim:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class PlayerGui : MonoBehaviour
05 {
06     [SerializeField]
07     Texture2D _crosshair;
08
09     Health _playerHealth;
10
11     void Start()
12     {
13         _playerHealth = GetComponent<Health>();
14     }
15
16     void OnGUI()
17     {
18         GUI.Label(new Rect(5,5,100,100), "Health: " + _playerHealth.ToString());
19
20         float x = (Screen.width - _crosshair.width) / 2;
21         float y = (Screen.height - _crosshair.height) / 2;
22         GUI.DrawTexture(new Rect(x, y, _crosshair.width, _crosshair.height), _crosshair);
23     }
24 }
25
```

Player'ın Health scriptini _playerHealth değişkeninde tutuyoruz ve OnGUI fonksiyonunda bu scriptteki ToString() fonksiyonunu kullanıyoruz.

Aslına bakarsanız ToString fonksiyonu özel bir fonksiyon. Kendisini çağırmak için o class'a erişmeniz yeterli. Yani class'a eriştikten sonra elle ToString() fonksiyonunu çağırmak zorunda değilsiniz, bu işlem otomatik yapılır (ToString'e has bir durum):

```
18         GUI.Label(new Rect(5,5,100,100), "Health: " + _playerHealth);
```

Oyunu çalıştırınca ekranın sol üstünde sağlığınızı görebilirsiniz.

Düşmanın Bize Saldırmasını Sağlamak

Zombi yakınımıza gelince bize saldırsın ve hasar versin.

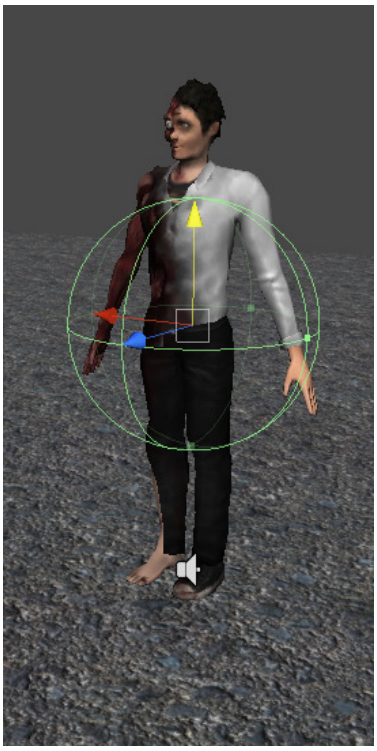
Yapacağımız şey şu: zombiye küre şeklinde bir collider (sphere collider) vereceğiz ve player bu collider'ın içinde olduğu süre boyunca hasar alacak.

GameObject > Create Empty ile "EnemyAttack” adında yeni bir obje oluşturup objeye Component > Physics > Sphere Collider ile bir Sphere Collider ekleyin.

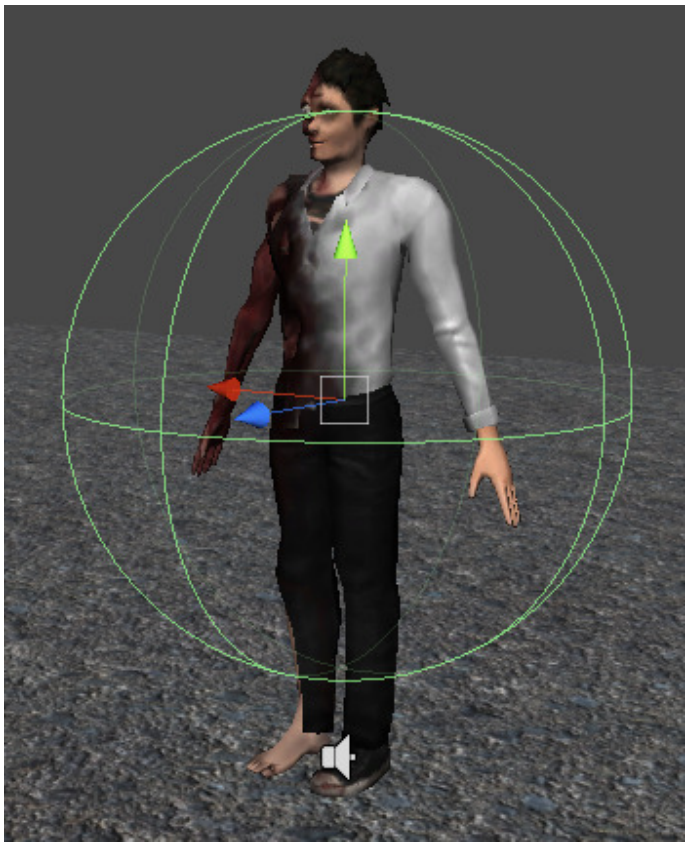
EnemyAttack objesini zombilerden tekinin child objesi yapın.

Sonrasında EnemyAttack'ın pozisyonunu (0, 0, 0), eğimini (rotation) da (0, 0, 0) ve boyutunu (scale) ise (1, 1, 1) yapın. (Kısa yolu: Transform'un sağındaki dişli ikona tıklayıp **Reset** deyin.)

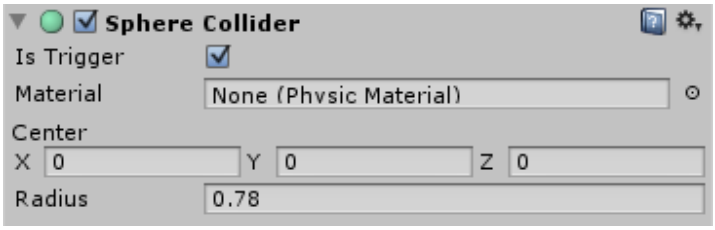
Şimdi kürenin merkezini göbek hizasına çıkarın:



Bu küre zombinin saldırı menzilini temsil ediyor. Şu an çok küçük olduğu için zombinin bize saldırması çok zor. Küreyi büyütelim ki zombinin menzili artsın, bize daha rahat saldırsın. Bunun için Inspector'da Sphere Collider'ın Radius (yarıçap) değerini 0.78 yapın.



Inspector'da bir de “Is Trigger” adında bir değer göreceksiniz. Onu işaretleyin. Bu seçeneği işaretlediğiniz zaman collider duvar etkisi yapmaz ve collider'ın içinden rahatça geçilebilir.



EnemyAttack objesinin seçili olduğundan emin olun ve objenin layer'ını “Ignore Raycast” olarak değiştirin. Player ateş ederken raycast sistemi kullanıyor ve biz bu raycast'i Enemy objesinin almasını istiyoruz, EnemyAttack objesinin değil. EnemyAttack'ın layer'ını "Ignore Raycast" yaptığımız için her ne kadar objede sphere collider olsa da raycast ışınları bu collider'ı yok sayacak ve böylece raycast yanlışlıkla EnemyAttack objesini vurmayacak.

Şimdi düşmanın hasar vermesini kodlayalım.

“EnemyAttack” adında yeni bir C# scripti oluşturun:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class EnemyAttack : MonoBehaviour
05 {
06     void OnTriggerStay(Collider other)
07     {
08         if (other.tag == "Player")
09         {
10             Debug.Log("we're colliding with player!");
11         }
12     }
13 }
14
```

OnTriggerStay fonksiyonu bir collider, "Is Trigger"ı işaretli başka bir collider ile temas halinde olduğu süre boyunca her frame'de çalıştırılır.

Yani Sphere Collider'ımıza bir başka collider temas ettiği müddetçe fonksiyon çalıştırılacak. Fonksiyonun içinde ise ilk önce temas eden objenin tag'ını kontrol ediyoruz ve eğer tag'ı "Player" ise Debug.Log fonksiyonu ile konsola yazı yazdırıyoruz.

Şimdi EnemyAttack scriptini EnemyAttack objesine verin. Sonrasında EnemyAttack'ın parent'ı olan Enemy objesini seçip Inspector'dan “Apply” butonuna basın ve böylece yaptığımız bu değişikliklerin tüm zombi prefab klonlarına uygulanmasını sağlamış olun.

Sıra geldi test etmeye! Bir zombi ile temas halinde olduğunuzda konsola mesaj yazdırılacak.

Player'a Hasar Vermek

Test başarılı olduğuna göre Debug.Log komutunu gerçek bir hasar verme kodu ile değiştirelim:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class EnemyAttack : MonoBehaviour
05 {
06     void OnTriggerStay(Collider other)
07     {
08         if (other.tag == "Player")
09         {
10             Health playerHealth = other.GetComponent<Health>();
11             playerHealth.Damage(1);
12         }
13     }
14 }
15
```

Oyunu çalıştırıp zombiler tarafından hasar alın. Sol üstteki sağlık azalacak ve öldüğünüz zaman ragdoll devreye girecek.

Herşey güzel duruyor ama player çok çabuk hasar alıyor ve ölüyor. Bunu çözmemiz lazım.

Hızlı Hasar Alma Sorununu Çözmek

OnTriggerStay'in collider ile temasta olduğumuz her frame'de çağrıldığından bahsettim. Eğer oyun 60 FPS'de (frames per second) oynuyorsa bunun anlamı fonksiyonun saniyede 60 kez çağrılması ve haliyle saniyede 60 hasar almamızdır!

Her frame'de 1 hasar almak yerine örneğin her 1 saniyede 5 hasar almak çok daha mantıklı:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class EnemyAttack : MonoBehaviour
05 {
06     float _nextTimeAttackIsAllowed = -1.0f;
07
08     void OnTriggerStay(Collider other)
09     {
10         if (other.tag == "Player" && Time.time >= _nextTimeAttackIsAllowed)
11         {
12             Health playerHealth = other.GetComponent<Health>();
13             playerHealth.Damage(5);
14             _nextTimeAttackIsAllowed = Time.time + 1.0f;
15         }
16     }
17 }
18
```

10. satırda Time.time kullanıyoruz. Bu değişkenin değeri oyun başladığında 0'dır ve her saniyede değeri 1 artar. “_nextTimeAttackIsAllowed” değişkeni zombinin hasar verebilmesi için Time.time'ın alması gereken minimum değeri depolar. Eğer Time.time bu değerden büyükse hasar verme kodlarını çalıştırıyoruz.

14. satırda, hasar aldıktan sonra “_nextTimeAttackIsAllowed” değişkeninin değerini Time.time değişkeninin değerinin 1 fazlası olarak ayarlıyoruz. Bunun anlamı da zombi şu andan (Time.time) bir saniye sonra bize tekrar saldırabilecek.

1 saniye ve 5 hasar değerlerini birer değişkenden çeksek daha güzel olur. Böylece bu değerleri kolayca istediğimiz gibi değiştirebiliriz:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class EnemyAttack : MonoBehaviour
05 {
06     float _nextTimeAttackIsAllowed = -1.0f;
07
08     [SerializeField]
09     float _attackDelay = 1.0f;
10
11     [SerializeField]
12     int _damageDealt = 5;
13
14     void OnTriggerStay(Collider other)
15     {
16         if (other.tag == "Player" && Time.time >= _nextTimeAttackIsAllowed)
17         {
18             Health playerHealth = other.GetComponent<Health>();
19             playerHealth.Damage(_damageDealt);
20             _nextTimeAttackIsAllowed = Time.time + _attackDelay;
21         }
22     }
23 }
24
```

Player'ın Ölümü İle İlgili Ufak Bir Sorun

Öldüğümüz zaman hareket etmeyi kesiyor ve animasyonu durduruyoruz ama hâlâ ateş edebiliyoruz. Bu sorunu çözmek için Health scriptini düzenleyin:

```
01 using UnityEngine;
02 using  System.Collections;
03
04 public class  Health : MonoBehaviour
05 {
06       [SerializeField]
07       int  _maximumHealth = 100;
08
09       int  _currentHealth = 0;
10
11       override public  string  ToString()
12       {
13             return  _currentHealth + " / "  + _maximumHealth;
14       }
15
16       void  Start()
17       {
18             _currentHealth = _maximumHealth;
19       }
20
21       public  void  Damage(int  damageValue)
22       {
23             _currentHealth -= damageValue;
24
25             if  (_currentHealth <= 0)
26             {
27                   Animation a = GetComponentInChildren<Animation>();
28                   a.Stop();
29
30                   Destroy(GetComponent<PlayerMovement>());
31                   Destroy(GetComponent<PlayerAnimation>());
32                   Destroy(GetComponent<RifleWeapon>());
33
34                   Destroy(GetComponent<EnemyMovement>());
35
36                   Destroy(GetComponent<CharacterController>());
37
38                   Ragdoll r = GetComponent<Ragdoll>();
39                   if  (r != null)
40                   {
41                         r.OnDeath();
42                   }
43             }
44       }
45 }
46
```

Ölünce RifleWeapon scriptini de objeden atıyoruz ve artık ateş etme imkanımız olmuyor.

Sağlığın Negatif Değer Almasını Önlemek

Farkedeceğiniz üzere sağlığımız negatif değer de alabiliyor ama bu biraz saçma duruyor sanki, değil mi? Çözümü ise basit:

```
01 using UnityEngine;
02 using  System.Collections;
03
04 public class  Health : MonoBehaviour
05 {
06       [SerializeField]
07       int  _maximumHealth = 100;
08
09       int  _currentHealth = 0;
10
11       override public  string  ToString()
12       {
13             return  _currentHealth + " / "  + _maximumHealth;
14       }
15
16       void  Start()
17       {
18             _currentHealth = _maximumHealth;
19       }
20
21       public  void  Damage(int  damageValue)
22       {
23             _currentHealth -= damageValue;
24
25             if  (_currentHealth < 0)
26             {
27                   _currentHealth = 0;
28             }
29
30             if  (_currentHealth == 0)
31             {
32                   Animation a = GetComponentInChildren<Animation>();
33                   a.Stop();
34
35                   Destroy(GetComponent<PlayerMovement>());
36                   Destroy(GetComponent<PlayerAnimation>());
37                   Destroy(GetComponent<RifleWeapon>());
38
39                   Destroy(GetComponent<EnemyMovement>());
40
41                   Destroy(GetComponent<CharacterController>());
42
43                   Ragdoll r = GetComponent<Ragdoll>();
44                   if  (r != null)
45                   {
46                         r.OnDeath();
47                   }
48             }
49       }
50 }
51
```

Düşmanın Ölse De Hâlâ Hasar Verebilmesini Durdurmak

Bir zombi öldüğünde ondan EnemyAttack scriptini silersek artık bize hasar veremeyecektir:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class Health : MonoBehaviour
05 {
06     [SerializeField]
07     int _maximumHealth = 100;
08
09     int _currentHealth = 0;
10
11     override public string ToString()
12     {
13         return _currentHealth + " / " + _maximumHealth;
14     }
15
16     void Start()
17     {
18         _currentHealth = _maximumHealth;
19     }
20
21     public void Damage(int damageValue)
22     {
23         _currentHealth -= damageValue;
24
25         if (_currentHealth < 0)
26         {
27             _currentHealth = 0;
28         }
29
30         if (_currentHealth == 0)
31         {
32             Animation a = GetComponentInChildren<Animation>();
33             a.Stop();
34
35             Destroy(GetComponent<PlayerMovement>());
36             Destroy(GetComponent<PlayerAnimation>());
37             Destroy(GetComponent<RifleWeapon>());
38
39             Destroy(GetComponent<EnemyMovement>());
40             Destroy(GetComponentInChildren<EnemyAttack>());
41
42             Destroy(GetComponent<CharacterController>());
43
44             Ragdoll r = GetComponent<Ragdoll>();
45             if (r != null)
46             {
47                 r.OnDeath();
48             }
49         }
50     }
51 }
52
```

EnemyAttack scriptinin bir child objede (EnemyAttack objesi) olduğunu hatırlayın. Bu sebepten ötürü scripte GetComponentInChildren ile erişiyoruz.

Ölünce Ekranda Mesaj Göstermek

Öldüğümüzde ekranda bunu belirten bir mesaj belirse daha güzel durur bence.

Project panelinde "TheGame" klasörünün içinde "Gui" adında yeni bir klasör oluşturun.

Şimdi Winrar arşivinin olduğu yerdeki "Gui" klasöründe yer alan "GameOverScreen.png" dosyasını projenize import edip Project panelinden "Gui" klasörünüzün içine atın.

Import ettiğiniz GameOverScreen asset'ini seçin ve Inspector'dan Texture Type'ı "GUI (Editor \ Legacy)" olarak değiştirin. Sonrasında “Apply” butonuna basın. Böyle yaptığımız zaman resmin arkaplanı görünmez oluyor (zaten önizleme penceresinden siz de görebilirsiniz).

Sırada Player öldüğünde ekranda bu resmi göstermek var. Ama önce Player'ın ölüp ölmediğini bilmemiz lazım. Health scriptini şöyle güncelleyin:

```
01 using UnityEngine;
02 using  System.Collections;
03
04 public class  Health : MonoBehaviour
05 {
06       [SerializeField]
07       int  _maximumHealth = 100;
08
09       int  _currentHealth = 0;
10
11       override public  string  ToString()
12       {
13             return  _currentHealth + " / "  + _maximumHealth;
14       }
15
16       public  bool  IsDead { get{ return  _currentHealth <= 0; } }
17
18       void  Start()
19       {
20             _currentHealth = _maximumHealth;
21       }
22
23       public  void  Damage(int  damageValue)
24       {
25             _currentHealth -= damageValue;
26
27             if  (_currentHealth < 0)
28             {
29                   _currentHealth = 0;
30             }
31
32             if  (_currentHealth == 0)
33             {
34                   Animation a = GetComponentInChildren<Animation>();
35                   a.Stop();
36
```

```
37                  Destroy(GetComponent<PlayerMovement>());
38                  Destroy(GetComponent<PlayerAnimation>());
39                  Destroy(GetComponent<RifleWeapon>());
40
41                  Destroy(GetComponent<EnemyMovement>());
42                  Destroy(GetComponentInChildren<EnemyAttack>());
43
44                  Destroy(GetComponent<CharacterController>());
45
46                  Ragdoll r = GetComponent<Ragdoll>();
47                  if (r != null)
48                  {
49                        r.OnDeath();
50                  }
51            }
52      }
53 }
54
```

IsDead ne bir değişken ne de bir fonksiyondur. Arada kalmış bir property'dir. (Bu property kavramını öğrenmek için Unity'nin resmî video tutorialine bakmanızı şiddetle tavsiye ederim: http://unity3d.com/learn/tutorials/modules/intermediate/scripting/properties)

Şimdi de “CombatGui” adında yeni bir C# scripti oluşturun:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class CombatGui : MonoBehaviour
05 {
06       Health _playerHealth;
07
08       void Start()
09       {
10             GameObject playerGameObject = GameObject.FindGameObjectWithTag("Player");
11             _playerHealth = playerGameObject.GetComponent<Health>();
12       }
13
14       void OnGUI()
15       {
16             if (_playerHealth.IsDead)
17             {
18                   // game over resmini burada ekrana çizdireceğiz
19             }
20       }
21 }
22
```

Player objesinin Health scriptini bir değişkende tutuyoruz. Ama önce Player objesine erişmemiz lazım (CombatGui scriptini Player'a değil başka bir objeye vereceğiz). Bunun için ise önce FindGameObjectWithTag ile "Player" tag'lı objeye, ardından onun Health scriptine erişiyoruz. OnGUI fonksiyonunda ise Player'ın ölüp ölmediğini sorguluyoruz.

Eğer Player ölmüşse ekranda az önce import ettiğimiz resmi gösterelim:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class CombatGui : MonoBehaviour
05 {
06     Health _playerHealth;
07
08     [SerializeField]
09     Texture2D _gameOverImage;
10
11     void Start()
12     {
13         GameObject playerGameObject = GameObject.FindGameObjectWithTag("Player");
14         _playerHealth = playerGameObject.GetComponent<Health>();
15     }
16
17     void OnGUI()
18     {
19         if (_playerHealth.IsDead)
20         {
21             GUI.DrawTexture(new Rect(0, 0, _gameOverImage.width, _gameOverImage.height),
22                                         _gameOverImage);
23         }
24     }
25 }
26
```

PlayerGui kodunda ekrana crosshair çizdirirken de benzer bir kod kullandığımız için açıklama yapmama gerek yok.

Artık scripti test edelim. "CombatGui" adında yeni bir empty gameobject oluşturun. CombatGui scriptini objeye verin. "Game Over Image" kısmına değer olarak “GameOverScreen” resmini verin.

Oyunu test edince game over yazısının ekranın ortasında değil sol üst noktasında gösterildiğini göreceksiniz. Çünkü resmin çizdirileceği Rect'in X ve Y koordinatları (0, 0). Bunu çözmek için crosshair'de kullandığımız "resmi ortalama" metodunu kullanalım:

```
                                                    .
                                                    .
                                                    .
17     void OnGUI()
18     {
19         if (_playerHealth.IsDead)
20         {
21             float x = (Screen.width - _gameOverImage.width) / 2;
22             float y = (Screen.height - _gameOverImage.height) / 2;
23             GUI.DrawTexture(new Rect(x, y, _gameOverImage.width, _gameOverImage.height),
24                                         _gameOverImage);
25         }
26     }
27 }
28
```

Ve işte sonuç karşınızda:

Özet Geçecek Olursak...

Bu bölümde düşmanların bize hasar verebilmesini sağladık. Bunun için Is Trigger'ı işaretli sphere collider kullandık. Ayrıca Time.time değişkenini de ilk defa kullanmış olduk. Çok iyi gidiyoruz!