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

Unity 3D İle Oyun Programlama

Bölüm 14: Ana Menü Oluşturmak

ÇEVİRİ: Süleyman Yasir KULA

http://yasirkula.com

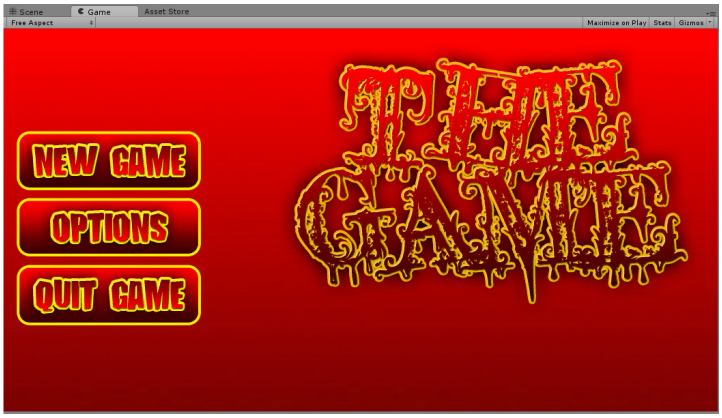
Ferdinand Joseph Fernandez

Chief Technological Officer, Dreamlords Digital Inc.

Admin and Co-founder, Unity Philippines Users Group

September 2011

Hemen her oyunda oyun açıldığı zaman karşımıza bir ana menü gelir. Buradan oyunu çalıştırabilir veya oyunun ses, görsel gibi ayarlarını değiştirebiliriz. Bizim ana menümüz şöyle duracak:



Görsel 14.1: Ana menüden bir görüntü

Gerekli Dosyaları Import Etmek

File-New Scene ile yeni bir sahne oluşturun. Sahneyi Scenes klasörünün içine “MainMenu” adıyla kaydedin.

Winrar arşivinin olduğu yerdeki "Gui" klasöründe yer alan şu dosyaları kendi projenizdeki "Gui" klasörüne import edin:

1. ButtonBg.png
2. ButtonBgPressed.png
3. ButtonNewGame.png
4. ButtonOptions.png
5. ButtonQuit.png
6. MainMenuBg.png
7. Title.png

Bu resim dosyalarını ana menüde kullanacağız (sizin Gui klasörünüzde henüz "GUISkin" asset'i yer almayacaktır, dert etmeyin. Dersin ilerleyen kısımlarında oluşturuyoruz onu). **Import ettiğiniz texture'lerin her birinin "Texture Type"ını "GUI (Editor \ Legacy)" yapın.**

"MainMenuGui" adında yeni bir C# scripti oluşturun:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MainMenuGui : MonoBehaviour
05 {
06     [SerializeField]
07     Texture2D _mainMenuBg;
08
09     void OnGUI()
10     {
11         GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
12     }
13 }
```

Tek yaptığımız şey arkaplan resmini ekranın genişliğini (Screen.width) ve yüksekliğini (Screen.height) kaplayacak şekilde ekrana çizdirmek.

MainMenuGui scriptini "Main Camera" objesine verin. Inspector'dan "Main Menu Bg"ye değer olarak MainMenuBg texture'sini verin.

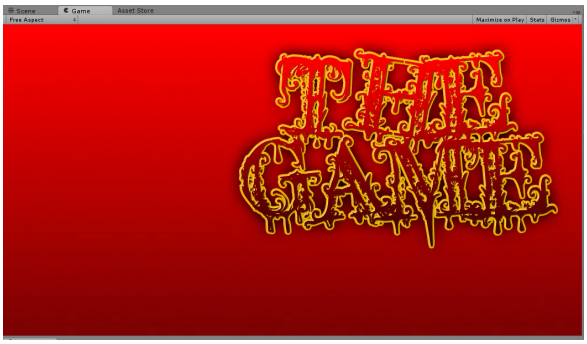
Oyunu çalıştırın. Basit bir kırmızı arkaplanın tüm ekranı kapladığını göreceksiniz.

Scripti biraz daha genişletelim:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MainMenuGui : MonoBehaviour
05 {
06     [SerializeField]
07     Texture2D _mainMenuBg;
08
09     [SerializeField]
10     Texture2D _title;
11
12     void OnGUI()
13     {
14         GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
15
16         GUI.DrawTexture(new Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
17                                  _title);
18     }
19 }
```

Yeni eklediğimiz kod ile ekranın sağ üstünde oyunun ismini barındıran texture'yi (title) çizdiriyoruz. Texture'yi çizdirdiğimiz Rect'in aldığı ilk parametreye dikkat edin: Screen.width - _title.width - 20. Bunun anlamı, texture ekranın sağ kenarından 20 pixel uzakta çizilecektir. İkinci parametrede ise direkt 20 değerini veriyoruz. Bunun anlamı da texture ekranın üst kenarından 20 pixel uzakta çizilecektir.

Şimdi Inspector'dan "Title"a değer olarak Gui klasöründeki "Title" asset'ini verip oyunu çalıştırın:

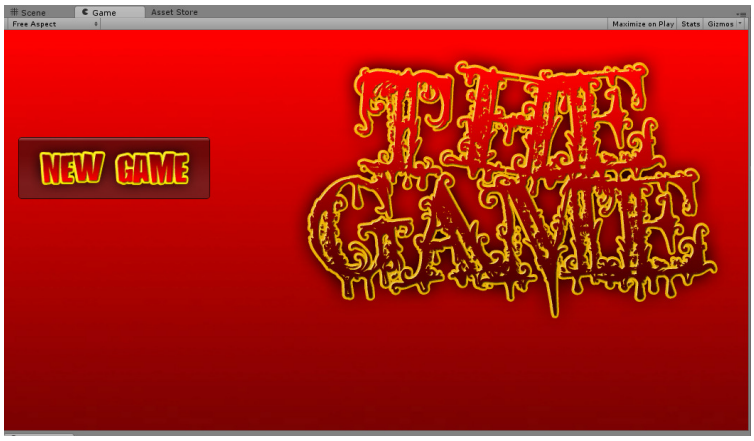


Sıra geldi butonları menüye eklemeye:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MainMenuGui : MonoBehaviour
05 {
06     [SerializeField]
07     Texture2D _mainMenuBg;
08
09     [SerializeField]
10     Texture2D _title;
11
12     [SerializeField]
13     Texture2D _newGameButton;
14
15     void OnGUI()
16     {
17         GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
18
19         GUI.DrawTexture(new Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
20                         _title);
21
22         if (GUI.Button(new Rect(20, 150, 270, _newGameButton.height), _newGameButton))
23         {
24         }
25     }
26 }
```

GUI.Button komutu sadece OnGUI fonksiyonunun içindeyken çalışır ve ekrana tıklanabilir bir buton çizdirmeye yarar. Eğer butona tıklanırsa fonksiyon true döndürür. Bu yüzden GUI.Button'u bir "if" koşulunun içine yazıyoruz; eğer butona tıklanırsa if'in içine yazdığımız kodlar çalışacak.

"ButtonNewGame" asset'ini "Main Menu Gui" component'indeki "New Game Button"a değer olarak verip oyunu çalıştırın. Ekranda tıklanabilir bir buton görünecek:



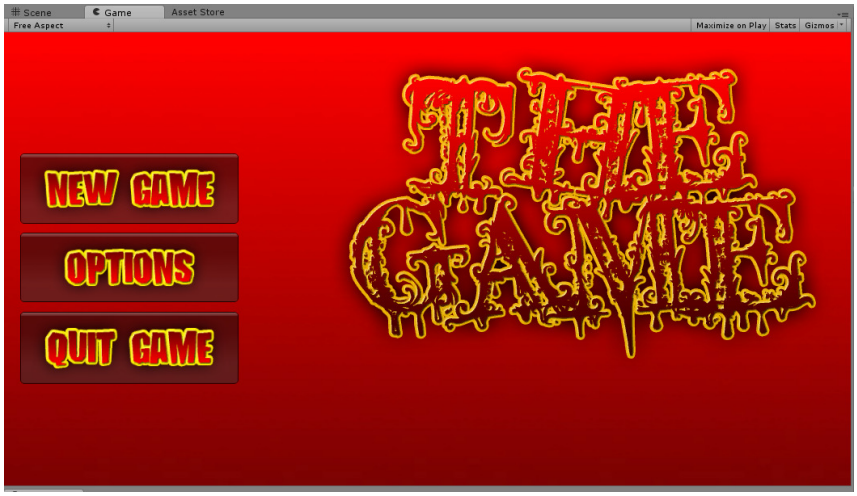
Hazır elimiz değmişken diğer butonları da menüye ekleyelim:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  MainMenuGui : MonoBehaviour
05 {
06       [SerializeField]
07       Texture2D _mainMenuBg;
08
09       [SerializeField]
10       Texture2D _title;
11
12       [SerializeField]
13       Texture2D _newGameButton;
14
15       [SerializeField]
16       Texture2D _optionsButton;
17
18       [SerializeField]
19       Texture2D _quitButton;
20
21       void  OnGUI()
22       {
23             GUI.DrawTexture(new  Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
24
25             GUI.DrawTexture(new  Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
26                                    _title);
27
28             if  (GUI.Button(new  Rect(20, 150, 270, _newGameButton.height), _newGameButton))
29             {
30             }
31
32             if  (GUI.Button(new  Rect(20, 150  + 88  + 10, 270, _optionsButton.height), _optionsButton))
33             {
34             }
35
36             if  (GUI.Button(new  Rect(20, 150  + (2*(88  + 10)), 270, _quitButton.height), _quitButton))
37             {
38             }
39       }
40 }
```

Rect'in ikinci parametresiyle oynayarak butonların farklı yüksekliklerde oluşturulmasını sağlıyoruz.

ÇEVİRMEN EKLEMESİ: 150 + 88 + 10'daki 150, 88 ve 10'un ne anlama geldiğini merak edebilirsiniz. 150, butonlar ile ekranın üst kenarı arasındaki boşluğun kaç pixel olduğunu belirliyor. İki buton arasındaki dikey boşluk ise (bir butonun yüksekliği) + (üstteki butonun alt kenarı ile alttaki butonun üst kenarı arasındaki boşluk) olarak hesaplanmakta. Bizim butonlarda kullandığımız texture'lerin yüksekliklerine Inspector'dan bakacak olursanız 87, 86 ve 89 pixel olduklarını göreceksiniz. Yazar bu sayıları ortalayıp "bir butonun yüksekliği"ni 88 pixel olarak ele almış. "üstteki butonun alt kenarı ile alttaki butonun üst kenarı arasındaki boşluk"un ise 10 pixel olmasını uygun görmüş.

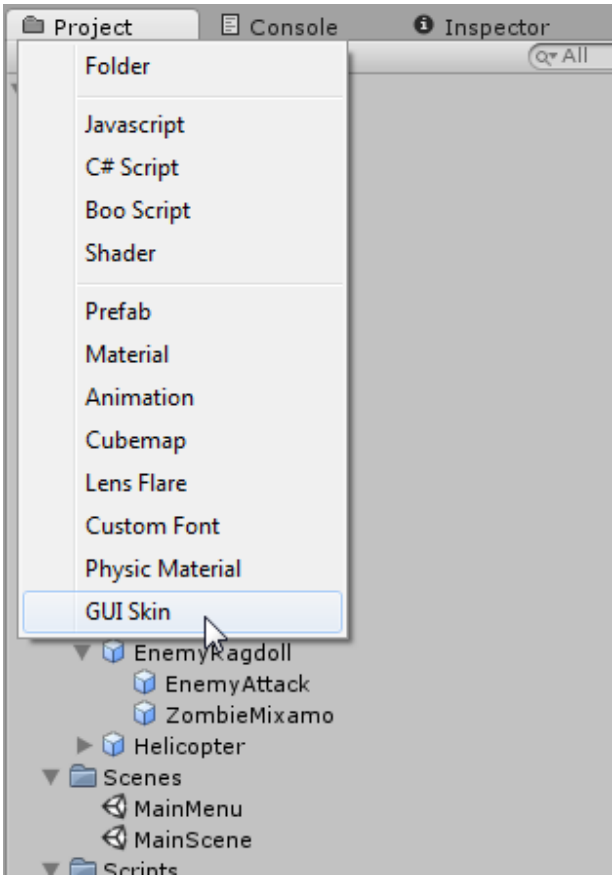
Inspector'daki "Options Button" ve "Quit Button"a uygun texture'leri değer olarak verip oyunu test edin:



Butonların Arkaplanını Değiştirmek

Butonların siyah arkaplanı menünün geri kalanıyla tarz olarak uyuşmuyor. Aslına bakarsanız bu siyah arkaplanın stilini değiştirmek bizim elimizde. Bunu yapmak için *GUI Skin* türünde bir asset'ten faydalanacağız. Bir GUI Skin ekranda çizdirilen GUI elemanlarının nasıl duracağını, kullanacakları fontu ve bunun gibi şeyleri belirler. Yani arayüzün stilini belirler.

Project panelinde "Gui" klasörünün içinde "Create-GUI Skin" yolunu izleyerek yeni bir GUI Skin oluşturun:

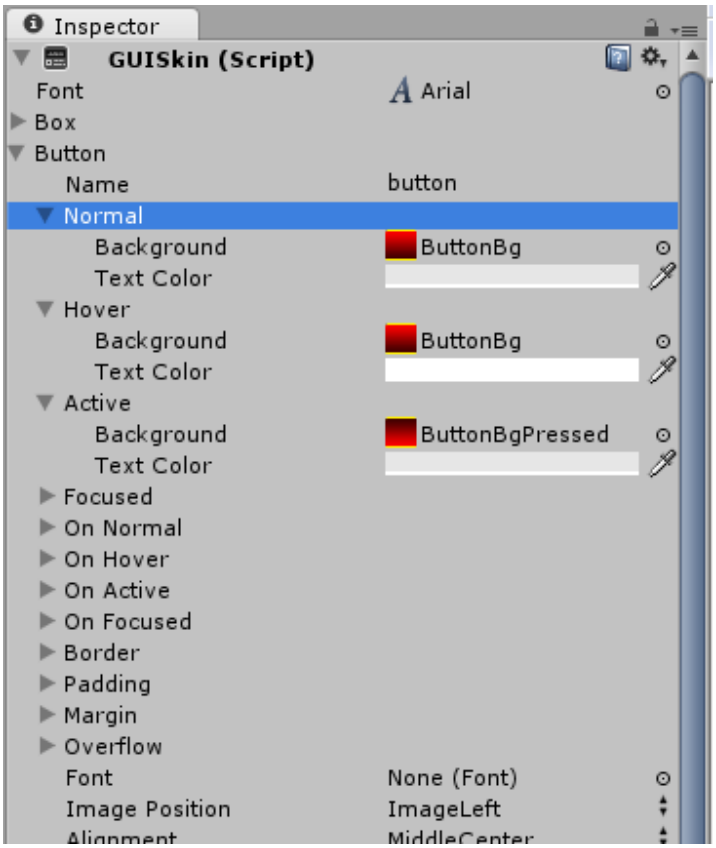


Oluşan dosyanın adını “GuiSkin” olarak değiştirin.

Oluşturduğunuz GUI Skin dosyasını seçin. Inspector'da çeşitli GUI elemanları için çeşitli ayarlar gözükecek. Buradaki her bir elemana bir "*GUI Style*" adı verilir.

Biz butonların stilini değiştirmek istiyoruz. Bunun için “Button”un yanındaki üçgene tıklayarak buton stilinin ayarlarının gözükmesini sağlayın.

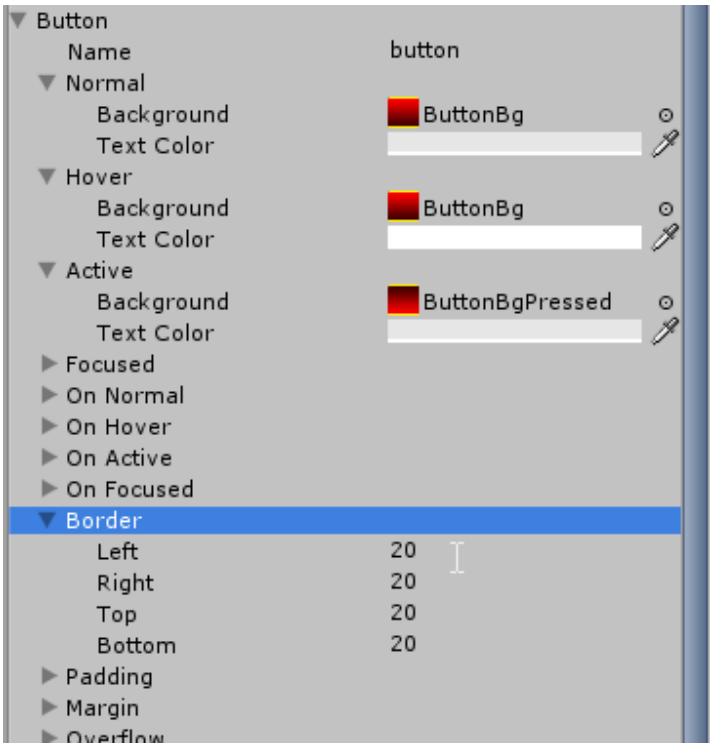
Buradaki “Normal”, “Hover” ve “Active”in "Background"una değer olarak şu texture'leri verin:



"Normal" butonun normal stilini, "Hover" fare imlecini butonun üzerine getirince butonun sahip olacağı stili ve "Active" de butona tıklayınca butonun sahip olacağı stili belirler. Bu stillerde yer alan “Background” değeri butonun o anda arkaplanında çizilecek resim dosyasını temsil eder.

Biz, buton normal dururken veya fare butonun üzerindeyken arkaplana "ButtonBg" texture'sini, butona tıklayınca ise arkaplana "ButtonBgPressed" texture'sini çizdireceğiz.

Şimdi “Border”a gelin. Butonun ebatları arkaplan texture'sinin ebatlarından büyükse arkaplanın nasıl genişletileceği burada belirlenir. Buradaki tüm değerleri 20 yapın:



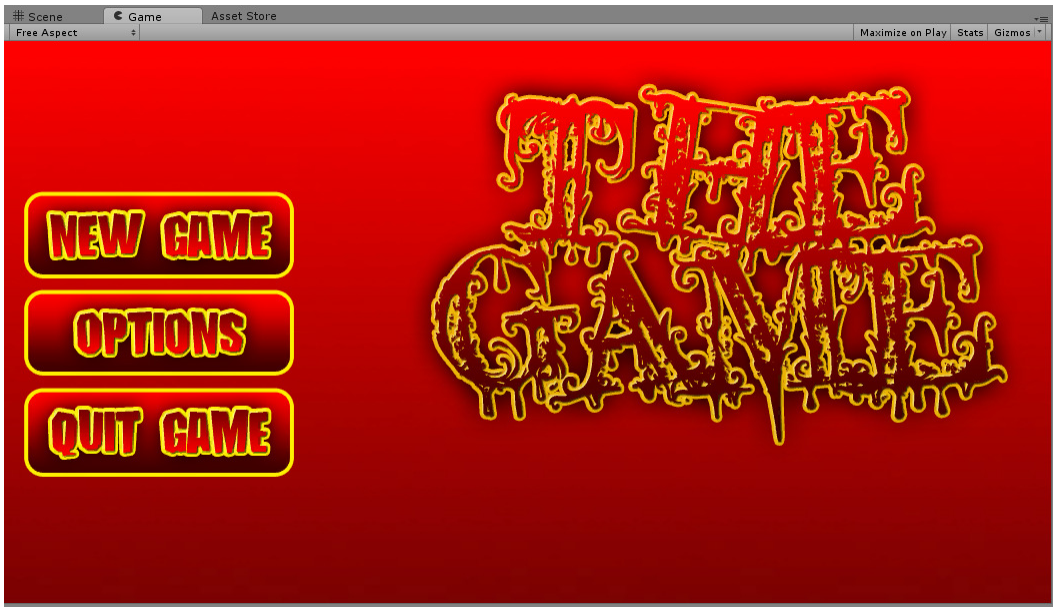
Artık bu GUI Skin'i kendi menümüzde kullanabiliriz. Bunun için MainMenuGui scriptini düzenleyelim:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MainMenuGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _mainMenuBg;
11
12     [SerializeField]
13     Texture2D _title;
14
15     [SerializeField]
16     Texture2D _newGameButton;
17
18     [SerializeField]
19     Texture2D _optionsButton;
20
21     [SerializeField]
22     Texture2D _quitButton;
23
24     void OnGUI()
25     {
26         GUI.skin = _guiSkin;
27
28         GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
29
30         GUI.DrawTexture(new Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
31                         _title);
```

```
32
33          if (GUI.Button(new Rect(20, 150, 270, _newGameButton.height), _newGameButton))
34          {
35          }
36
37          if (GUI.Button(new Rect(20, 150 + 88 + 10, 270, _optionsButton.height), _optionsButton))
38          {
39          }
40
41          if (GUI.Button(new Rect(20, 150 + (2*(88 + 10)), 270, _quitButton.height), _quitButton))
42          {
43          }
44      }
45 }
```

"GUI.skin = _guiSkin;" diyerek OnGUI fonksiyonunda oluşturduğumuz arayüz (GUI) elemanlarının _guiSkin değişkeninde tutulan GUI Skin'deki stilleri kullanmasını sağlıyoruz.

Inspector'dan "Gui Skin"e oluşturduğunuz GUI Skin asset'ini değer olarak verip oyunu çalıştırın:

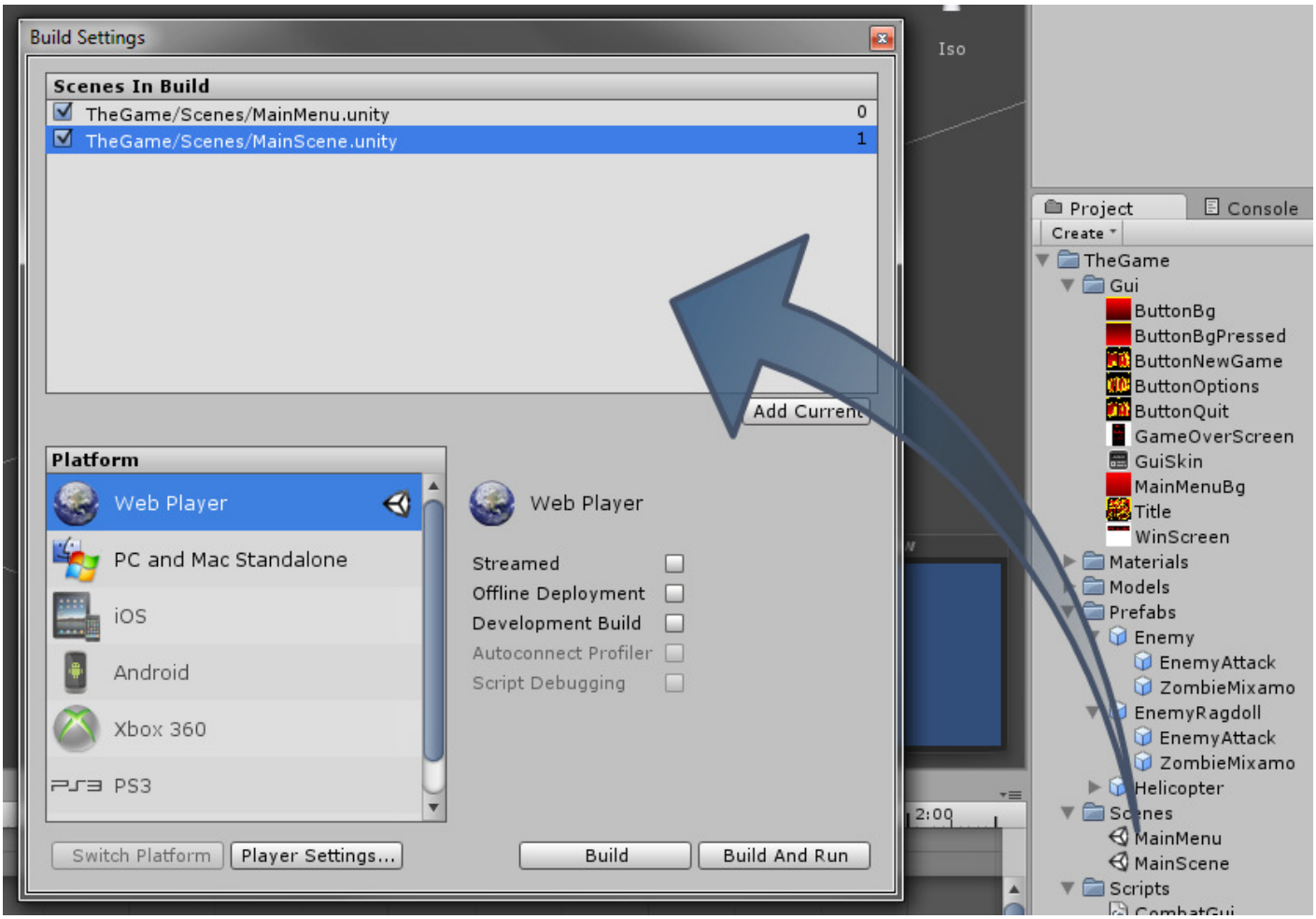


New Game Butonuna Basınca Yapılacaklar

Oyuncu New Game butonuna basınca oyunun olduğu sahneye (scene)(level) geçiş yapmamız lazım.

İlk önce Unity'e, oluşturduğumuz sahnelerden hangilerinin oyuna gerçekten dahil olduğunu söylememiz lazım. File > Build Settings... yolunu izleyin.

Build Settings penceresi gelecek. Burada “Scenes In Build” adında boş bir liste var. Yapmanız gereken Scenes klasöründe yer alan MainMenu ile MainScene'i bu listeye sürüklemek (Listedeki ilk scene, build edilen oyun başladığında açılacak olan scene'dir. Bu yüzden eğer listenin başında MainMenu değil de MainScene varsa MainMenu'yü sürükle-bırak yaparak listenin başına alın):



MainMenuGui scriptini açıp şu eklemeyi yapın:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  MainMenuGui : MonoBehaviour
05 {
06         [SerializeField]
07         GUISkin _guiSkin;
08
09         [SerializeField]
10         Texture2D _mainMenuBg;
11
12         [SerializeField]
13         Texture2D _title;
14
15         [SerializeField]
16         Texture2D _newGameButton;
17
18         [SerializeField]
19         Texture2D _optionsButton;
20
21         [SerializeField]
22         Texture2D _quitButton;
```

```
23
24      void OnGUI()
25      {
26          GUI.skin = _guiSkin;
27
28          GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
29
30          GUI.DrawTexture(new Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
31                                  _title);
32
33          if (GUI.Button(new Rect(20, 150, 270, _newGameButton.height), _newGameButton))
34          {
35              Application.LoadLevel("MainScene");
36          }
37
38          if (GUI.Button(new Rect(20, 150 + 88 + 10, 270, _optionsButton.height), _optionsButton))
39          {
40          }
41
42          if (GUI.Button(new Rect(20, 150 + (2*(88 + 10)), 270, _quitButton.height), _quitButton))
43          {
44          }
45      }
46 }
```

Application.LoadLevel fonksiyonu başka bir scene'e geçiş yapmaya yarar. Mevcut scene'deki objeler bu esnada silinirler; yani diğer bölüme geçmezler. Bu fonksiyonun çalışması için fonksiyona parametre olarak girilen scene'in Build Settings'teki "Scenes In Build" listesinde yer alması lazım. Biz de zaten bu yüzden her iki scene'i de listeye ekledik.

Oyunu çalıştırıp “New Game”e basın ve mevcut scene'in değiştiğine kendi gözlerinizle tanık olun!

Quit Butonuna Basınca Yapılacaklar

Oyunu sonlandıracağız. Bunu yapmak çok basit:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MainMenuGui : MonoBehaviour
05 {
06      [SerializeField]
07      GUISkin _guiSkin;
08
09      [SerializeField]
10      Texture2D _mainMenuBg;
11
12      [SerializeField]
13      Texture2D _title;
14
15      [SerializeField]
16      Texture2D _newGameButton;
17
18      [SerializeField]
19      Texture2D _optionsButton;
```

```
20
21      [SerializeField]
22      Texture2D _quitButton;
23
24      void  OnGUI()
25      {
26            GUI.skin = _guiSkin;
27
28            GUI.DrawTexture(new  Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
29
30            GUI.DrawTexture(new  Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
31                                    _title);
32
33            if  (GUI.Button(new  Rect(20, 150, 270, _newGameButton.height), _newGameButton))
34            {
35                  Application.LoadLevel("MainScene");
36            }
37
38            if  (GUI.Button(new  Rect(20, 150  + 88  + 10, 270, _optionsButton.height), _optionsButton))
39            {
40            }
41
42            if  (GUI.Button(new  Rect(20, 150  + (2*(88  + 10)), 270, _quitButton.height), _quitButton))
43            {
44                  Application.Quit();
45            }
46      }
47 }
```

Bu kod editörde bir işe yaramaz ama oyunu örneğin .EXE olarak build ederseniz işte o zaman işe yarar.

Ayarlar (Options) Menüsünü Oluşturmak

Geriye sadece Options menüsünü oluşturmak kaldı. Bu menüden oyunun zorluğunu (difficulty) ve ses düzeyini (volume) değiştirebileceğiz. Yapacağımız Options menüsünden bir görüntü:

"OptionsGui” adında yeni bir C# scripti oluşturun:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  OptionsGui : MonoBehaviour
05 {
06       [SerializeField]
07       GUISkin _guiSkin;
08
09       void  OnGUI()
10       {
11            GUI.skin = _guiSkin;
12       }
13 }
```

MainMenuGui'de yaptığımız gibi, GUI elemanlarının oluşturduğumuz GUI Skin'deki stilleri kullanmasını sağlıyoruz.

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  OptionsGui : MonoBehaviour
05 {
06       [SerializeField]
07       GUISkin _guiSkin;
08
09       void  OnGUI()
10       {
11            GUI.skin = _guiSkin;
12
13            GUI.Box(new  Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
14       }
15 }
```

Ardından ekrana, kenarlardan 15 pixel boşluk kalacak şekilde, basit bir kutucuk çizdiriyoruz.

Scripti "Main Camera"ya verin, "Gui Skin"e değerini atayın ve oyunu test edin.

Kutucuğu siyah yarı-saydam bir arkaplana sahip olarak göreceksiniz; bizim bu siyah arkaplanı değiştirmemiz lazım:

Project panelinden GuiSkin asset'ini seçin. Inspector'dan “Box”ı genişletin. Buradaki "Normal" isimli stil bizi ilgilendiren şey. "Hover" ve "Active" önemli değil çünkü bir kutucuğa tıklamak gibi birşey mümkün değil.

ButtonBg texture'sini, "Normal" stilinin "Background"una atayın. İşlem tamamlanınca “Border”daki değerlerin hepsini 20 yapın; tıpkı "Button"da yaptığımız gibi...

Oyunu çalıştırın. Kutucuk çok daha güzel görünecek:

Şimdi Winrar arşivinin ordaki "Gui" klasöründen projenizdeki "Gui" klasörüne şu dosyaları import edin:

1. ButtonOk.png
2. CheckBoxChecked.png
3. CheckBoxUnchecked.png
4. SliderBg.png
5. SliderThumb.png
6. TextEasy.png
7. TextHard.png
8. TextNormal.png
9. TitleDifficulty.png
10. TitleOptions.png
11. TitleSound.png

Tüm bu texture'lerin de "Texture Type"ını “GUI (Legacy \ Editor)” yapmayı unutmayın.

Ardından şu kodu OptionsGui scriptine ekleyin:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  OptionsGui : MonoBehaviour
05 {
06       [SerializeField]
07       GUISkin _guiSkin;
08
09       [SerializeField]
10       Texture2D _title;
11
12       void  OnGUI()
13       {
14             GUI.skin = _guiSkin;
15
```

```
16              GUI.Box(new  Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
17
18              GUI.DrawTexture(new  Rect(20, 20, _title.width, _title.height), _title);
19      }
20 }
```

Inspector'dan "Title"a değer olarak "TitleOptions" resmini verin ve oyunu test edin:

Şimdi kodu şöyle güncelleyin:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  OptionsGui : MonoBehaviour
05 {
06      [SerializeField]
07      GUISkin _guiSkin;
08
09      [SerializeField]
10      Texture2D _title;
11
12      [SerializeField]
13      Texture2D _ok;
14
15      void  OnGUI()
16      {
17              GUI.skin = _guiSkin;
18
19              GUI.Box(new  Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
20
21              GUI.DrawTexture(new  Rect(20, 20, _title.width, _title.height), _title);
22
23              if  (GUI.Button(new  Rect(Screen.width - 30  - _ok.width - 10, Screen.height - 30  - _ok.height
24                      - 10, _ok.width, _ok.height), _ok))
25              {
26              }
27      }
28 }
```

Options menüsünden geri Main Menu'ye dönmek için OK yazan bir buton oluşturuyoruz. Rect koordinatlarını öyle veriyoruz ki ekran çözünürlüğü ne olursa olsun OK butonu hep ekranın sağ altında yer alıyor.

Inspector'dan yeni değişkene değer olarak "ButtonOk" texture'sini verin:

Şimdi Options menüsünden gerçekten de Main Menu'ye dönmek için gerekli kodu yazalım:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  OptionsGui : MonoBehaviour
05 {
06       [SerializeField]
07       GUISkin _guiSkin;
08
09       [SerializeField]
10       Texture2D _title;
11
12       [SerializeField]
13       Texture2D _ok;
14
15       static  bool  _isOpen = true;
16
17       void  OnGUI()
18       {
19            if  (!_isOpen)
20            {
21                 return;
22            }
23
24            GUI.skin = _guiSkin;
25
26            GUI.Box(new  Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
27
28            GUI.DrawTexture(new  Rect(20, 20, _title.width, _title.height), _title);
29
30            if  (GUI.Button(new  Rect(Screen.width - 30  - _ok.width - 10, Screen.height - 30  - _ok.height
31                 - 10, _ok.width, _ok.height), _ok))
32            {
33                 Close();
34            }
35       }
36
37       void  Close()
38       {
```

```
39              _isOpen = false;
40          }
41  }
```

Options menüsünde olup olmadığımızı belirleyen _isOpen adında bir değişken oluşturduk. OK butonuna basınca değerini false yapıyoruz. Değişkenin değeri false ise 21. satırdaki "return;" komutu çalıştırılıyor ve OnGUI'nin 21. satırdan sonraki satırları çalıştırılmıyor (Options menüsü ekrana çizdirilmiyor).

Oyunu çalıştırıp OK butonuna basın. Options menüsünün kapanması lazım.

Options Menüsünü Ana Menüden Açmak

Şu anda oyunun başında Options menüsü aktif durumda. Ama asıl olması gereken ana menünün ekranda gözükmesi ve buradaki Options butonuna basınca Options menüsünün belirmesi.

Bunu düzeltelim. OptionsGui scriptinde şu ufak değişikliği yapın:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  OptionsGui : MonoBehaviour
05 {
06      [SerializeField]
07      GUISkin _guiSkin;
08
09      [SerializeField]
10      Texture2D _title;
11
12      [SerializeField]
13      Texture2D _ok;
14
15      static  bool  _isOpen = false;
16
17      void  OnGUI()
18      {
19          if  (!_isOpen)
20          {
21              return;
22          }
23
24          GUI.skin = _guiSkin;
25
26          GUI.Box(new  Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
27
28          GUI.DrawTexture(new  Rect(20, 20, _title.width, _title.height), _title);
29
30          if  (GUI.Button(new  Rect(Screen.width - 30  - _ok.width - 10, Screen.height - 30  - _ok.height
31              - 10, _ok.width, _ok.height), _ok))
32          {
33              Close();
34          }
35      }
36
37      void  Close()
38      {
39          _isOpen = false;
```

```
40      }
41 }
```

Artık oyunun başında Options menüsü aktif olmayacak (çünkü _isOpen'ın başlangıç değeri false).

Ana menüdeyken Options butonuna basınca Options menüsünün gelmesini sağlayalım. OptionsGui'ye şu eklemeyi yapın:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class OptionsGui : MonoBehaviour
05 {
06      [SerializeField]
07      GUISkin _guiSkin;
08
09      [SerializeField]
10      Texture2D _title;
11
12      [SerializeField]
13      Texture2D _ok;
14
15      static bool _isOpen = false;
16
17      static public void Open()
18      {
19          _isOpen = true;
20      }
21
22      void OnGUI()
23      {
24          if (!_isOpen)
25          {
26              return;
27          }
28
29          GUI.skin = _guiSkin;
30
31          GUI.Box(new Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
32
33          GUI.DrawTexture(new Rect(20, 20, _title.width, _title.height), _title);
34
35          if (GUI.Button(new Rect(Screen.width - 30 - _ok.width - 10, Screen.height - 30 - _ok.height
36              - 10, _ok.width, _ok.height), _ok))
37          {
38              Close();
39          }
40      }
41
42      void Close()
43      {
44          _isOpen = false;
45      }
46 }
```

Open fonksiyonu ile Options menüsünü aktif hale getirebiliyoruz. Tek yapmamız gereken Options butonuna tıklayınca bu fonksiyonu çağırmak. MainMenuGui scriptini güncelleyin:

```
01 using  UnityEngine;
02 using  System.Collections;
03
04 public class  MainMenuGui : MonoBehaviour
05 {
06       [SerializeField]
07       GUISkin _guiSkin;
08
09       [SerializeField]
10       Texture2D _mainMenuBg;
11
12       [SerializeField]
13       Texture2D _title;
14
15       [SerializeField]
16       Texture2D _newGameButton;
17
18       [SerializeField]
19       Texture2D _optionsButton;
20
21       [SerializeField]
22       Texture2D _quitButton;
23
24       void  OnGUI()
25       {
26             GUI.skin = _guiSkin;
27
28             GUI.DrawTexture(new  Rect(0, 0, Screen.width, Screen.height), _mainMenuBg);
29
30             GUI.DrawTexture(new  Rect(Screen.width - _title.width - 20, 20, _title.width, _title.height),
31                                    _title);
32
33             if  (GUI.Button(new  Rect(20, 150, 270, _newGameButton.height), _newGameButton))
34             {
35                   Application.LoadLevel("MainScene");
36             }
37
38             if  (GUI.Button(new  Rect(20, 150  + 88  + 10, 270, _optionsButton.height), _optionsButton))
39             {
40                   OptionsGui.Open();
41             }
42
43             if  (GUI.Button(new  Rect(20, 150  + (2*(88  + 10)), 270, _quitButton.height), _quitButton))
44             {
45                   Application.Quit();
46             }
47       }
48 }
```

Oyunu çalıştırın. Options butonunun istendiği gibi çalışması lazım.

Options Menüsünün Arkasında Kalan Butonlara İstemsiz Tıklamak

Oyunu çalıştırın ve Options menüsündeyken, ana menüde yer alan New Game butonunun olduğu yere tıklayın. Her ne kadar New Game butonu ekranda gözükmese de ona tıklayabiliyoruz çünkü bu butonun önünü GUI.Box ile kapatmamız onu tıklamamıza bir engel teşkil etmiyor.

Yapmamız gereken şu: ana menüdeki butonlara tıklayınca sadece ve sadece eğer ki options menüsü kapalıysa o butonun yapması gerekenleri yerine getirmeliyiz. Bunun için de _isOpen'ın false olup olmadığına bakmalıyız.

Maalesef _isOpen private bir değişken. Ona başka scriptlerden erişmek için bir property oluşturalım:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class OptionsGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _title;
11
12     [SerializeField]
13     Texture2D _ok;
14
15     static bool _isOpen = false;
16     static public bool IsOpen { get{ return _isOpen; } }
17
18     static public void Open()
19     {
20         _isOpen = true;
21     }
22
23     void OnGUI()
24     {
25         if (!_isOpen)
26         {
27             return;
28         }
29
30         GUI.skin = _guiSkin;
31
32         GUI.Box(new Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
33
34         GUI.DrawTexture(new Rect(20, 20, _title.width, _title.height), _title);
35
36         if (GUI.Button(new Rect(Screen.width - 30 - _ok.width - 10, Screen.height - 30 - _ok.height
37             - 10, _ok.width, _ok.height), _ok))
38         {
39             Close();
40         }
41     }
42
43     void Close()
44     {
45         _isOpen = false;
46     }
47 }
```

Ardından MainMenuGui'yi güncelleyelim:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class MainMenuGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _mainMenuBg;
11
12     [SerializeField]
13     Texture2D _title;
14
15     [SerializeField]
16     Texture2D _newGameButton;
17
18     [SerializeField]
19     Texture2D _optionsButton;
20
21     [SerializeField]
22     Texture2D _quitButton;
23
24     void OnGUI()
25     {
26         GUI.skin = _guiSkin;
27
28         GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height),
29                         _mainMenuBg);
30
31         GUI.DrawTexture(new Rect(Screen.width - _title.width - 20, 20, _title.
32                         width, _title.height), _title);
33
34         if (GUI.Button(new Rect(20, 150, 270, _newGameButton.height),
35             _newGameButton) && !OptionsGui.IsOpen)
36         {
37             Application.LoadLevel("MainScene");
38         }
39
40         if (GUI.Button(new Rect(20, 150 + 88 + 10, 270, _optionsButton.height),
41             _optionsButton) && !OptionsGui.IsOpen)
42         {
43             OptionsGui.Open();
44         }
45
46         if (GUI.Button(new Rect(20, 150 + (2*(88 + 10)), 270, _quitButton.
47             height), _quitButton) && !OptionsGui.IsOpen)
48         {
49             Application.Quit();
50         }
51     }
52 }
```

Zorluk (Difficulty) Ayarı Eklemek

Bunu checkbox (işaret kutucuğu) kullanarak yapacağız.

OptionsGui scriptini düzenleyin:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class OptionsGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _title;
11
12     [SerializeField]
13     Texture2D _ok;
14
15     [SerializeField]
16     Texture2D _titleDifficulty;
17
18     [SerializeField]
19     Texture2D[] _difficulties;
20
21     static int _selectedDifficulty = 0;
22
23     static bool _isOpen = false;
24     static public bool IsOpen { get{ return _isOpen; } }
25
26     static public void Open()
27     {
28         _isOpen = true;
29     }
30
31     void OnGUI()
32     {
33         if (!_isOpen)
34         {
35             return;
36         }
37
38         GUI.skin = _guiSkin;
39
40         GUI.Box(new Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
41
42         GUI.DrawTexture(new Rect(20, 20, _title.width, _title.height), _title);
43
44         GUI.DrawTexture(new Rect(50, 150, _titleDifficulty.width,
45                                  _titleDifficulty.height), _titleDifficulty);
46
47         _selectedDifficulty = GUI.SelectionGrid(new Rect(100, 250, 280, 400),
48                                                 _selectedDifficulty, _difficulties, 1);
49
50         if (GUI.Button(new Rect(Screen.width - 30 - _ok.width - 10, Screen.
51             height - 30 - _ok.height - 10, _ok.width, _ok.height), _ok))
52         {
```

```
53                  Close();
54              }
55      }
56
57      void  Close()
58      {
59              _isOpen = false;
60      }
61 }
```

44. satırda, üzerinde "Difficulty:” yazan bir texture'yi ekrana çizdiriyoruz.

47. satırda ise Selection Grids adında yeni bir GUI elemanından faydalanıyoruz. Bu GUI elemanı bir dizi checkbox'tan oluşur ve bu checkbox'lar arasından aynı anda sadece birisi seçili olabilir. Bu da zorluk seçmek için ideal birşey.

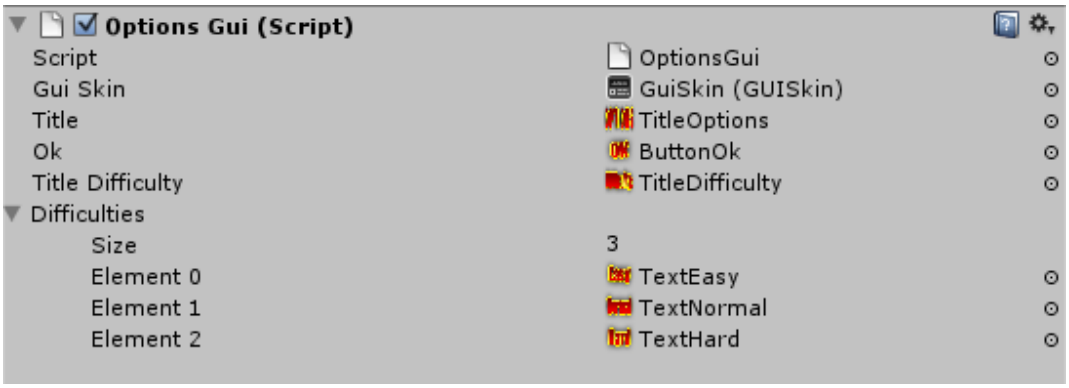
GUI.SelectionGrid fonksiyonu seçili olan checkbox'ın sırasını döndürür. Eğer ilk checkbox seçili ise 0, ikinci seçili ise 1, üçüncü seçili ise 2 döndürür ve bu böyle devam eder. Biz bu döndürülen değeri _selectedDifficulty değişkenine veriyoruz.

GUI.SelectionGrid'in ikinci parametresine de _selectedDifficulty yazıyoruz. Fonksiyon bu parametre sayesinde ekrandaki checkbox'lardan hangisine tik işareti koyacağını anlar.

Üçüncü parametre olarak checkbox'ların yanında yer alacak olan Texture'leri barındıran bir array (dizi) giriyoruz. Eğer istersek checkbox'ların yanında texture göstermez, sadece bir string yazdırırız. Bunu yapmak isterseniz 3. parametreyi bir string array'i ile değiştirmelisiniz.

Son parametremiz ise checkbox listesinin kaç sütundan oluşacağını belirler. Değerini 1 verdiğimiz için 1 sütundan oluşur ve checkbox'lar alt alta dizilirler. Ama diyelim ki GUI.SelectionGrid'e 4 elemanlı bir array'i parametre olarak verip sütun sayısını 2 yapsaydık o zaman bir satırda 2 checkbox gözükürdü (yani checkbox'lar 2x2'lik bir matris şeklinde gözükürdü).

Options Gui component'indeki değişkenlere resimdeki gibi değerlerini verin:



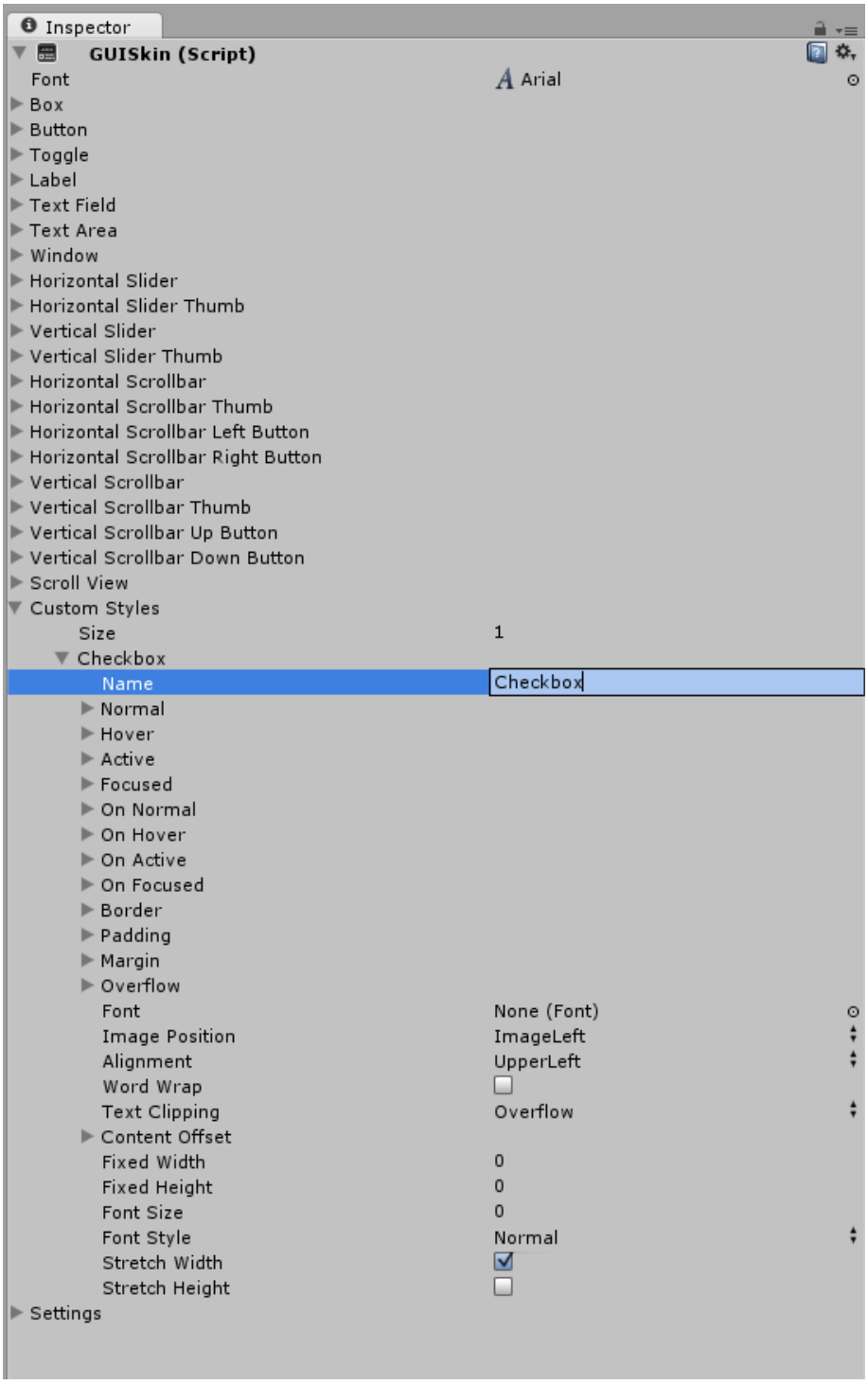
Oyunu test edin:

Checkbox'larımız pek bir çirkin duruyorlar.

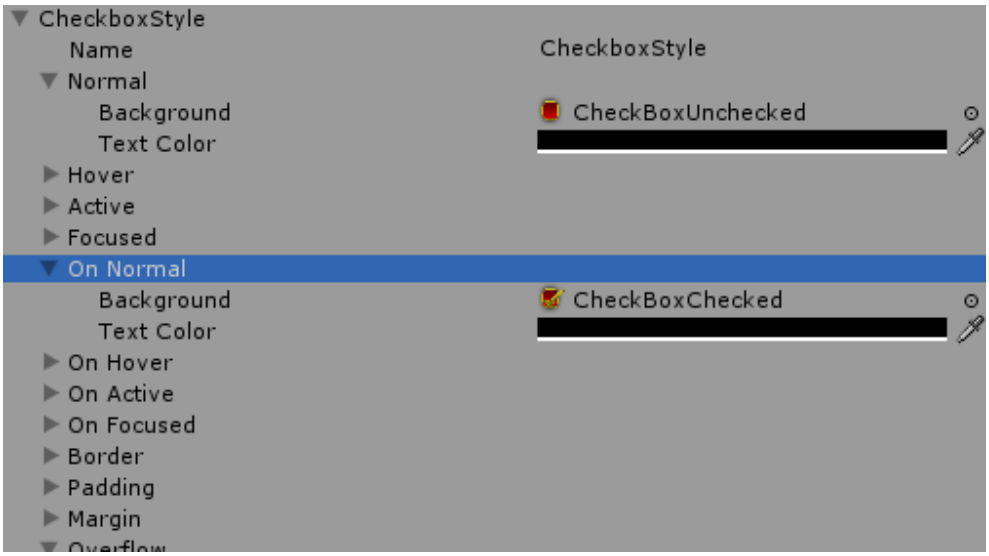
Unity'nin, checkbox'ları ekrana çizdirirken kullanacağı arkaplan texture'lerini biz belirlemeliyiz ve bunun için de yeni bir stile ihtiyacımız var.

Project panelinden GuiSkin'i seçin.

“Custom Styles”ın içindeki “Element 0”ı genişletin. Bu stilin ismini (Name) “CheckboxStyle” yapın (alttaki resimde Checkbox diye duruyor ama siz "CheckboxStyle" yapın).



Şimdi “Normal” ve “On Normal” kısımlarını genişletin ve “Background”larına şöyle değer atayın:



"Normal" stili checkbox seçili değilken kullanılırken "On Normal" stili checkbox seçiliyken kullanılır.

Artık GUI.SelectionGrid'in, oluşturduğumuz bu stili kullanmasını sağlayalım. OptionsGui scriptini açıp şu şekilde güncelleyin:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class OptionsGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _title;
11
12     [SerializeField]
13     Texture2D _ok;
14
15     [SerializeField]
16     Texture2D _titleDifficulty;
17
18     [SerializeField]
19     Texture2D[] _difficulties;
20
21     static int _selectedDifficulty = 0;
22
23     static bool _isOpen = false;
24     static public bool IsOpen { get{ return _isOpen; } }
25
26     static public void Open()
27     {
28         _isOpen = true;
29     }
30
31     void OnGUI()
32     {
33         if (!_isOpen)
```

```
34              {
35                  return;
36              }
37
38              GUI.skin = _guiSkin;
39
40              GUI.Box(new  Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
41
42              GUI.DrawTexture(new  Rect(20, 20, _title.width, _title.height), _title);
43
44              GUI.DrawTexture(new  Rect(50, 150, _titleDifficulty.width,
45                                  _titleDifficulty.height), _titleDifficulty);
46
47              _selectedDifficulty = GUI.SelectionGrid(new  Rect(100, 250, 280, 400),
48                                                  _selectedDifficulty, _difficulties, 1,
49                                                  "CheckboxStyle");
50
51              if  (GUI.Button(new  Rect(Screen.width - 30  - _ok.width - 10, Screen.
52                  height - 30  - _ok.height - 10, _ok.width, _ok.height), _ok))
53              {
54                  Close();
55              }
56      }
57
58      void  Close()
59      {
60              _isOpen = false;
61      }
62 }
```

GUI.SelectionGrid'in en sonuna, kullanmak istediğimiz stilin ismini parametre olarak giriyoruz. Bu kadar basit!

Oyunu test ederseniz sonucun tam da istediğimiz gibi olmadığını görebilirsiniz:

Checkbox'lar halen tam düzgün gözükmüyor. GUI stilinde bazı ayarlar yapmamız lazım.

> **Bilgilendirme**
>
> GUI Skin'deki ayarları değiştirirken oyunu durdurmak zorunda değilsiniz. Yaptığınız değişikliklerin sonuçlarını anında gözlemleyebilirsiniz!

GUI Skin asset'inin “CheckboxStyle”ındaki “Overflow” kısmını genişletin ve oradaki değerleri şöyle değiştirin:

- Left: 0
- Right: -72
- Top: 0
- Bottom: 0

“Fixed Width” ve “Fixed Height” değerlerini ise şu şekilde değiştirin:

- Fixed Width: 192
- Fixed Height: 97

Checkbox'ların boyutu düzelecek. Geriye yazıyı checkbox'un üzerinden taşımak kaldı:

“Content Offset”i bulun ve değerini şöyle güncelleyin:

- X: 77
- Y: 0

Son olarak “Alignment”ı “MiddleLeft” yapın:

Sonuç tek kelimeyle harika!

Ses Düzeyini (Volume) Ayarlamak

Henüz oyunumuzda ses yok ama ileride olacak. Bu seslerin şiddetini ise şimdiden ayarlayabileceğiz; ne kadar da şanslıyız!

OptionsGui scriptini düzenleyin:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class OptionsGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _title;
11
12     [SerializeField]
13     Texture2D _ok;
14
15     [SerializeField]
16     Texture2D _titleDifficulty;
17
18     [SerializeField]
19     Texture2D[] _difficulties;
20
21     [SerializeField]
22     Texture2D _titleSound;
23
24     static int _selectedDifficulty = 0;
25
26     static float _soundVolume = 1.0f;
27
28     static bool _isOpen = false;
29     static public bool IsOpen { get{ return _isOpen; } }
30
31     static public void Open()
32     {
33         _isOpen = true;
34     }
35
36     void OnGUI()
37     {
38         if (!_isOpen)
39         {
40             return;
41         }
42
43         GUI.skin = _guiSkin;
44
45         GUI.Box(new Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
46
47         GUI.DrawTexture(new Rect(20, 20, _title.width, _title.height), _title);
48
```

```
49
50              GUI.DrawTexture(new  Rect(50, 150, _titleDifficulty.width,
51                                        _titleDifficulty.height), _titleDifficulty);
52
53              _selectedDifficulty = GUI.SelectionGrid(new  Rect(100, 250, 280, 400),
54                                                      _selectedDifficulty, _difficulties, 1,
55                                                      "CheckboxStyle");
56
57
58              GUI.DrawTexture(new  Rect(440, 150, _titleSound.width, _titleSound.
59                                        height), _titleSound);
60
61              _soundVolume = GUI.HorizontalSlider(new  Rect(490, 250, Screen.width-490-
62                                        50, 20), _soundVolume, 0.0f, 1.0f);
63
64              if  (GUI.Button(new  Rect(Screen.width - 30  - _ok.width - 10, Screen.
65                    height - 30  - _ok.height - 10, _ok.width, _ok.height), _ok))
66              {
67                    Close();
68              }
69        }
70
71        void  Close()
72        {
73              _isOpen = false;
74        }
75  }
```

GUI.HorizontalSlider fonksiyonu, yatay eksende (horizontal) kaydırılabilir bir slider oluşturmaya yarar. Ses düzeyini _soundVolume değişkeninde tutuyoruz. Eğer değeri 0.0 ise hiç ses yokken 1.0 ise sesler %100'lük bir şiddetle çalıyorlar. 0.5 ise de sesler %50 şiddette çalıyorlar.

GUI.HorizontalSlider'ın ilk parametresi ekrandaki konumu (Rect). Bu Rect'in genişliği (üçüncü parametresi) Screen.width'ten faydalanıyor ve böylece slider ekranın çözünürlüğü ne olursa olsun ekranın sağ kenarına kadar uzanıyor.

GUI.HorizontalSlider'ın 2. parametresi mevcut ses şiddetini değer olarak alıyor ve ona göre slider'daki çubuğu ekranda çizdiriyor.

3. ve 4. parametreler de slider'ın döndürdüğü minimum ve maksimum değerleri belirliyor. Eğer çubuk slider'ın en solundaysa 0.0, en sağındaysa 1.0 döndürülmesini sağlıyoruz.

Inspector'dan “Title Sound”a değer olarak "TitleSound" texture'sini verin ve oyunu çalıştırın:

Siyah slider stili öteki kırmızı stillerin yanında garip duruyor. Bunun için slider'ın stilini de değiştirelim. GUI Skin'deki “Horizontal Slider” stili slider'ın arkaplanını değiştirmeye yararken “Horizontal Slider Thumb” ise slider'ın üzerindeki çubuğun stilini değiştirmeye yarar.

“Horizontal Slider”ın “Normal”indeki “Background”ı “SliderBg” olarak değiştirin.

“Horizontal Slider Thumb”ın “Normal”, “Hover” ve "Active”indeki “Background” değerlerini ise “SliderThumb” olarak değiştirin.

Sonrasında “Horizontal Slider”da şu değişiklikleri yapın:

- Overflow
  - Left: 0
  - Right: 0
  - Top: 25
  - Bottom: -25
- Fixed Height: 55

Overflow'u bu şekilde değiştirince slider biraz yukarı kayıyor. Fixed Height'ı 55 yapınca ise slider'ın hep 55 pixel yükseliğinde olmasını sağlıyoruz.

“Horizontal Slider Thumb” stilinde ise şu değişiklikleri yapın:

- Overflow
  - Left: 0
  - Right: 0
  - Top: 28
  - Bottom: -28
- Fixed Width: 62
- Fixed Height: 62

Artık çok şık bir slider'ımız var:

Options Menüsünde Yaptığınız Değişiklikleri Kaydetmek

Şu anda oyunu kapatıp tekrar açınca ayarlar menüsündeki ayarlarınız resetleniyor. Bu ayarları (zorluk ve ses düzeyi) cihaza kaydetmeli ve oyundan çıkıp yeniden girince ayarların en son halinde kalmasını

sağlamalıyız. Bunu, Unity'nin PlayerPrefs class'ı ile başarabiliriz.

Değerleri Kaydetmek

OptionsGui scriptine şu kodu ekleyin:

```
                                                    .
                                                    .
                                                    .
36      void OnGUI()
37      {
38          if (!_isOpen)
39          {
40              return;
41          }
42
43          GUI.skin = _guiSkin;
44
45          GUI.Box(new Rect(15, 15, Screen.width - 30, Screen.height - 30), "");
46
47          GUI.DrawTexture(new Rect(20, 20, _title.width, _title.height), _title);
48
49
50          GUI.DrawTexture(new Rect(50, 150, _titleDifficulty.width,
51                                   _titleDifficulty.height), _titleDifficulty);
52
53          _selectedDifficulty = GUI.SelectionGrid(new Rect(100, 250, 280, 400),
54                                                  _selectedDifficulty, _difficulties, 1,
55                                                  "CheckboxStyle");
56          PlayerPrefs.SetInt("Difficulty", _selectedDifficulty);
57
58
59          GUI.DrawTexture(new Rect(440, 150, _titleSound.width, _titleSound.
60                                   height), _titleSound);
61
62          _soundVolume = GUI.HorizontalSlider(new Rect(490, 250, Screen.width-490-
63                                   50, 20), _soundVolume, 0.0f, 1.0f);
64          PlayerPrefs.SetFloat("SoundVolume", _soundVolume);
65
66
67          if (GUI.Button(new Rect(Screen.width - 30 - _ok.width - 10, Screen.
68              height - 30 - _ok.height - 10, _ok.width, _ok.height), _ok))
69          {
70              Close();
71          }
72      }
73
74      void Close()
75      {
76          _isOpen = false;
77      }
78 }
```

56. satırda PlayerPrefs.SetInt fonksiyonunu kullanıyoruz. Bu fonksiyon cihaza bir tamsayı (int) değerini kaydetmeye yarar. İlk parametresi kaydedilecek değere sonradan erişmek için kullanacağımız bir isimdir: "Difficulty".

Bu değer cihazın harddiskinde kaydedilmekte ve bu işlem Unity tarafından otomatik olarak yapılmakta. Yani sizin harddisktteki dosyalara erişmekle, onları düzenlemekle vb. hiç uğraşmanız gerekmiyor.

64. satırda da PlayerPrefs.SetFloat ile float türündeki ses şiddetini "SoundVolume" adı altında cihaza kaydediyoruz.

Oyuna Başlayınca Değerleri Cihazdan Okumak

Oyun başlayınca ayarlar menüsündeki değerleri cihazdan okumalıyız ki ayarlar son bıraktığımız halde dursun.

OptionsGui scriptini güncelleyin:

```
01 using UnityEngine;
02 using System.Collections;
03
04 public class OptionsGui : MonoBehaviour
05 {
06     [SerializeField]
07     GUISkin _guiSkin;
08
09     [SerializeField]
10     Texture2D _title;
11
12     [SerializeField]
13     Texture2D _ok;
14
15     [SerializeField]
16     Texture2D _titleDifficulty;
17
18     [SerializeField]
19     Texture2D[] _difficulties;
20
21     [SerializeField]
22     Texture2D _titleSound;
23
24     static int _selectedDifficulty = 0;
25
26     static float _soundVolume = 1.0f;
27
28     static bool _isOpen = false;
29     static public bool IsOpen { get{ return _isOpen; } }
30
31     static public void Open()
32     {
33         _isOpen = true;
34     }
35
36     void Start()
37     {
38         _selectedDifficulty = PlayerPrefs.GetInt("Difficulty", 1);
39         _soundVolume = PlayerPrefs.GetFloat("SoundVolume", 1.0f);
40     }
41
42     void OnGUI()
43     {
```

```
44          if (!_isOpen)
45          {
46              return;
47          }
48
49          GUI.skin = _guiSkin;
                                                    .
                                                    .
                                                    .
```

Nasıl SetInt ve SetFloat cihaza bir değeri kaydetmeye yarıyorsa GetInt ve GetFloat da cihazda kayıtlı bir değeri okumaya yarar. Fonksiyona ilk parametre olarak cihazda kaydedilen değerin ismi girilir. İkinci parametre olarak da eğer ki cihazda bu isimde bir değer kaydedilmemişse fonksiyonun döndüreceği değer girilir.

Özet Geçecek Olursak...

Artık bir ana menümüz var. Bu bölümde nasıl GUI Skin'den ve stillerden faydalanabileceğinizi gördünüz. Ayrıca cihaza bir değeri kaydedip sonradan bu değeri geri okumayı da gördünüz.