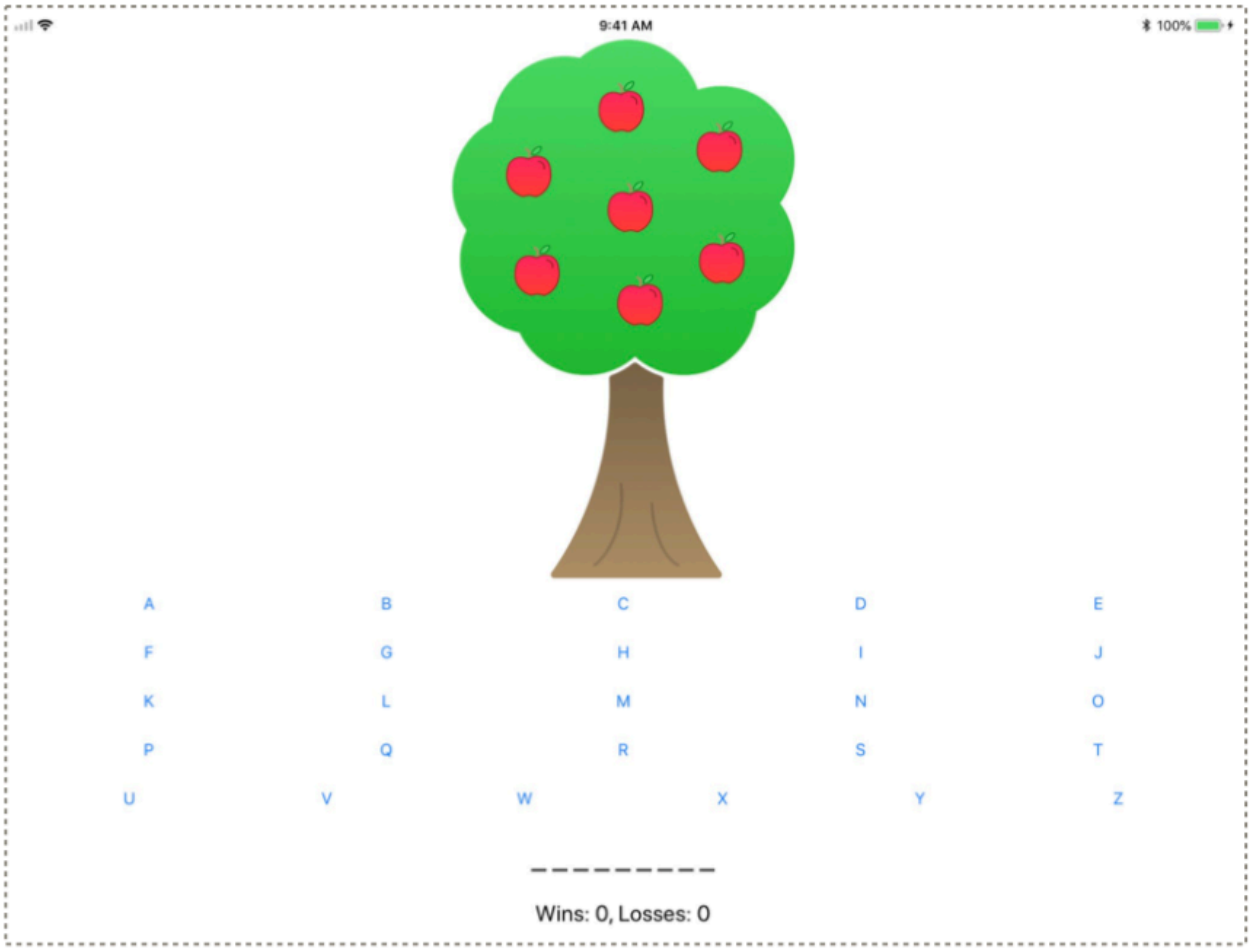
Proje: Adam Asmaca Oyunu ("Apple Pie")

Bu projenin amacı, kullanıcının birtakım kelimeleri tahmin etmesini sağlayan bir adam asmaca oyunu kodlamaktır.



İlk olarak **“Single View Application”** şablonunu kullanarak **“Apple Pie”** adında yeni bir proje açın ve cihaz türünü **"iPad"** olarak ayarlayın.

Adım 1: Arayüz Oluşturma

1- Main.storyboard dosyasını açın.

2- “Interface Builder” kısmında “View as” seçeneklerini yatay konum ve iPad olarak belirleyin.

3- Arayüze bir Vertical Stack View ekleyin.

4- Vertical Stack View elemanına, ekranın kenarı ile Stack View'ın kenarları (sağ, sol, üst, alt) arasındaki boşluk sıfır olarak şekilde dört tane constraint ekleyin.

5- Stack View elemanının içerisinde bir Image View ekleyin.

6- Image View elemanının "Content Mode" seçeneğini "Aspect Fit" olarak değiştirin.

7- Öğrenci kaynakları klasöründeki elma ağacı resmini Assets klasörüne ekleyin.

8- Eklediğiniz Image View'ın resmini elma ağacı resmi olarak belirleyin. Böylece Storyboard üzerinde arayüzünüzü daha iyi bir şekilde görebileceksiniz.

9- Büyük Stack View elemanının içerisinde ImageView elemanının altına bir Vertical Stack View daha ekleyin. Bu Vertical Stack View, harflerin butonlarını barındıracak.

10- Yeni eklediğiniz Stack View elemanına, değeri 220 olan bir yükseklik (height) constraint'i ekleyin.

11- Yeni eklediğiniz Stack View elemanının "Distribution" özelliğini "Fill Equally" olarak ayarlayın. Bu sayede, ekleyeceğiniz butonların yükseklikleri eşit olacak.

12- Yeni eklediğiniz Stack View elemanının içerisinde bir Horizontal Stack View ekleyin. Bu Horizontal Stack View, içerisine eklediğiniz butonları yatay olarak sıralayacak.

13- Horizontal Stack View elemanının "Distribution" özelliğini "Fill Equally" olarak ayarlayın.

14- Horizontal Stack View elemanının içerisine bir buton ekleyin.

15- Horizontal Stack View elemanının içerisine aynı şekilde dört buton daha ekleyin. Bunu kopyala-yapıştır ile yapabilirsiniz. Toplamda Horizontal Stack View elemanınızın içerisinde 5 tane buton yer almalı.

16- Şimdi Horizontal Stack View elemanını seçin, kopyalayın ve dikey eksende altı tane Horizontal Stack View olana kadar yapıştırın. Bu işlemin sonunda ekranda toplam 30 tane buton yer almalı.

17- Sonuncu Horizontal Stack View elemanın içerisinden bir butonu silin. (Alfabede 29 tane harf var!)

18- Buton metinlerini her butonda alfabenin bir harfi yazacak şekilde ayarlayın.

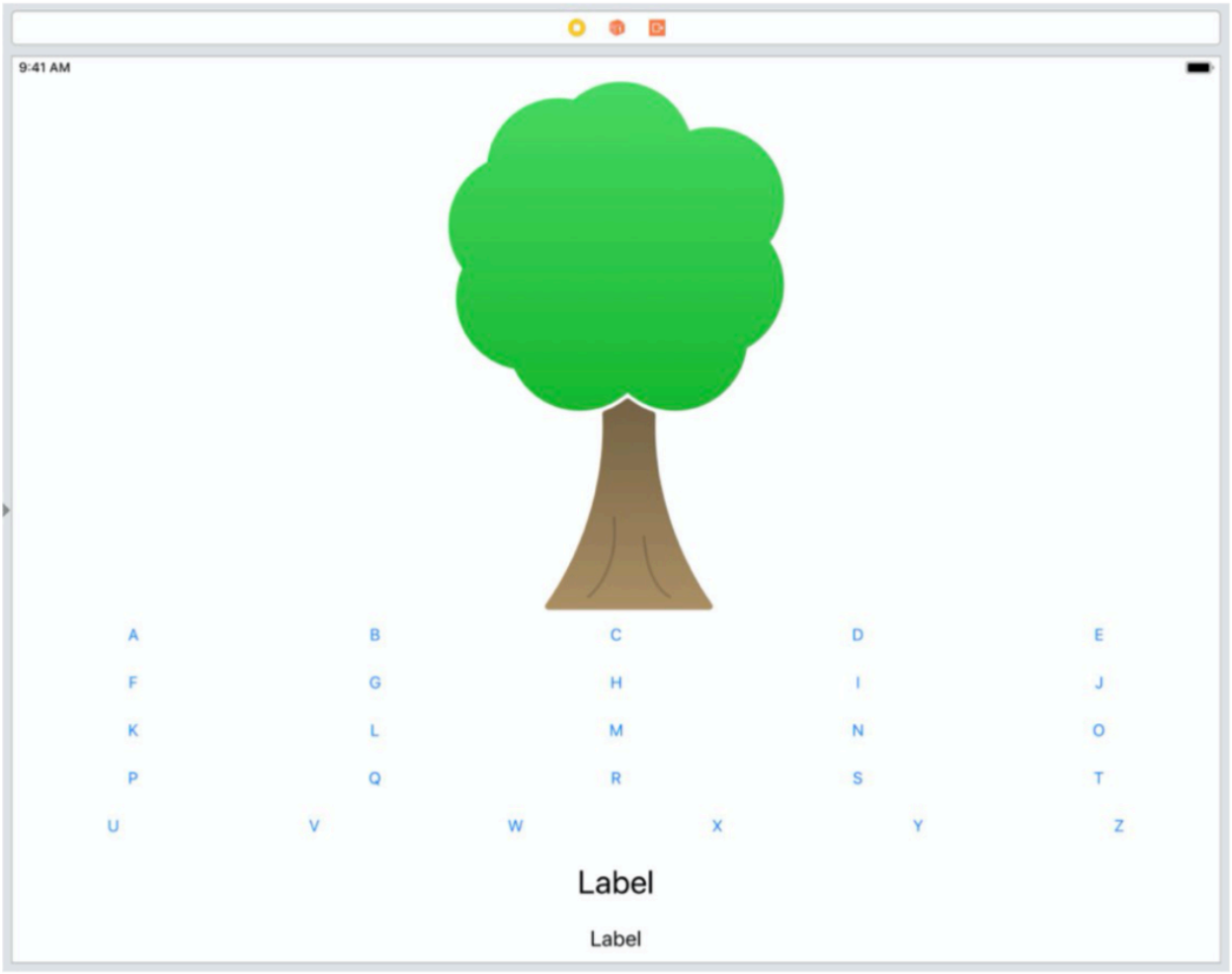
19- Butonların bulunduğu Stack View elemanının altına iki tane Label elemanı ekleyin.

20- Üstteki Label elemanının font boyutunu 30 pt, alttaki Label elemanının font boyutunu ise 20 pt olarak ayarlayın.

21- Üstteki Label elemanına değeri 60 olan bir yükseklik (height) constraint'i ekleyin.

22- Alttaki Label elemanına değeri 60 olan bir yükseklik (height) constraint'i ekleyin.

Arayüzü oluşturmayı bitirdiğinizde, ekranınınız aşağıdakine benzemeli:



Adım 2: IBOutlet Bağlantılarını Oluşturma

1- Sağ üst köşedeki editör ekleme butonuna tıklayın ve yeni editör ekranının üst kısmından "ViewController.swift" dosyasını seçin.

2- Storyboard ekranındaki Image View elementine tıklayın.

3- Kntrl-sürükle yöntemi ile Image View elemanı için **"agacImageView"** adında bir IBOutlet oluşturun.

4- Aynı şekilde iki Label için de IBOutlet'ler oluşturun. Üstteki Label elemanını **"cozumLabel**, alttaki Label elemanını ise **"puanLabel"** olarak isimlendirin.

5- Şimdi, bütün harfleri içerisinde tutacak olan bir IBOutlet oluşturacaksınız. Bunun için, ekrandaki ilk harfe tıklayıp kntrl-sürükle yöntemini kullanın. Ancak çıkan ekranda bu sefer "Connection Type" seçeneğini "Outlet Collection" olarak değiştirin. Bu yeni IBOutlet'i **"harfButonlari"** olarak isimlendirin.

6- "harfButonlari" adlı IBOutlet'in sol tarafındaki yuvarlak butona basılı tutun ve Storyboard üzerindeki diğer bir butonun üzerine sürükleyip bırakın. Bu işlemi Storyboard ekranındaki tüm harf butonları için tekrarlayın.

7- Butonlar, tıklandıklarından çalışacak olan bir IBAction'a da bağlanmalı. Her bir buton için ayrı bir IBAction oluşturmak yerine, her butonun çağıracağı tek bir IBAction oluşturacaksınız. Bunun için öncelikle ilk butona tıklayıp ViewController.swift dosyasına kntrl-sürükleyin. Ancak bu sefer çıkan ekranda "Connection Type" seçeneğini "Action" olarak değiştirin. IBAction'ı **"butonTiklandi"** olarak isimlendirin ve "Type" seçeneğini "UIButton" olarak ayarlayın. Bunu ayarlamanız, IBAction fonksiyonunun içerisinde butonların başlık metinlerine erişebilmenizi sağlayacak.

8- IBAction'ın kenarındaki yuvarlak butona tıklayıp sürükleyerek diğer tüm butonları da IBAction'a bağlayın.

9- Adam asmaca oyununda, her harfi sadece bir kez kullanabilirsiniz. Bu nedenlere, butonlara bir kez tıklandıktan sonra kullanılamaz hale gelmeleri gerekiyor. Bunu yapmak için, oluşturduğunuz IBAction'ın içerisinde `sender.isEnabled` özelliğini `false` değerine eşitleyin:

```
@IBAction func butonTiklandi(_ sender: UIButton) {
  sender.isEnabled = false
}
```

Adım 3: Oyuna Başlangıç

1- İlk olarak kullanıcılarınızın içerisinden kelime tahmin edebileceği bir kelime havuzu hazırlamalısınız. Bunun için ViewController sınıfının başında **"kelimeListesi"** isimli bir değişken tanımlayın.

2- "kelimeListesi" değişkenini çeşitli kelimelerden oluşan bir diziye eşitleyin. Kelimelerinizde sadece küçük harf kullanın.

ÖRNEK

```
var kelimeListesi = ["elma", "kuş", "büyüleyici", "parlak", "böcek",
"kodlama"]
```

3- "kelimeListesi" değişkeninin altında **"tahminSayisi"** isimli bir sabit tanımlayın. Bu sabit, kelime başına kaç tahmin yapılabileceğini tutacak. Toplam yedi farklı elma ağacı resmi bulunduğundan, bu değer 1 ila 7 arasında olmalı.

ÖRNEK

```
let tahminSayisi = 7
```

4- "toplamDogru" ve **"toplamYanlis"** isminde, başlangıç değerleri 0 olan iki tane değişken tanımlayın. Bu değişkenler, kazanılan ve kaybedilen turların sayısını tutacak.

5- "yeniTur" isimli, parametreleri olmayan bir metod tanımlayın. Bu metod, yeni bir tur başladığında gerekli değerleri sıfırlayacak.

6- "viewDidLoad" fonksiyonun içerisinde **"yeniTur"** metodunu çağırın.

7- File > New > File... menüsünden "Oyun.swift" isimli yeni bir dosya oluşturun.

8- "Oyun.swift" dosyasının içerisinde **"Oyun"** adında bir struct oluşturun.

9- "Oyun" struct'ının içerisinde, **"kelime"** isimli bir String özellik tanımlayın.

10- "kelime" özelliğinin altına, **"kalanTahminSayisi"** isimli bir Int özellik tanımlayın.

11- "ViewController" sınıfının içerisinde, **"simdikiOyun"** adında, "Game" türünden, varsayılan değeri olmayan bir özellik tanımlayın.

ÖRNEK

```
var simdikiOyun: Game!
```

12- "yeniTur" metodunda, **"kelimeListesi"** dizinin ilk değerini `removeFirst()` metodu ile alın ve **"yeniKelime"** isimli bir sabite eşitleyin.

13- "simdikiOyun" değişkeninin değerini, bir "Game" nesnesine eşitleyin. Bu nesneyi memberwise initializer ile oluşturun. **"kelime"** parametresine **"yeniKelime"** sabitini, **"kalanTahminSayisi"** parametresine ise **"tahminSayisi"** sabitini verin.

14- "ViewController" sınıfının içerisinde **"arayuzuGuncelle"** isimli parametresiz bir metod tanımlayın. Bu metod, her turun sonunda arayüzü güncelleyecek.

15- "yeniTur" metodunun son satırında **"arayuzuGuncelle"** metodunu çağırın.

16- "arayuzuGuncelle" metodunun içerisinde, **"puanLabel"** elemanının "text" özelliğini **"toplamDogru"** ve **"toplamYanlis"** değerlerini gösterecek şekilde güncelleyin.

17- "arayuzuGuncelle" metodunun içerisinde, **"agacImageView"** elemanının "image" özelliğini "Tree (x)" adındaki UIImage olacak şekilde güncelleyin. Buradaki (x) değeri, **"kalanTahminSayisi"** değerine eşit olmalı.

Uygulamanızı çalıştırın. Alt kısımdaki Label elemanları, yeni bir turun başladığını gösteriyor olmalı. Tebrikler! 🎉

Adım 4: Oyun Durumunu Güncelleme

1- "butonTiklandi" metodunun içerisinde "sender" nesnesinin "title" özelliğini tutan, **"harfString"** isimli bir sabit tanımlayın.

2- "harf" isimli bir sabit tanımlayın. Bu sabitin değeri, **"harfString"** sabitinin "lowercased" halini içeren bir "Character" nesnesi olacak.

ÖRNEK

```
@IBAction func butonTiklandi(_ sender: UIButton) {
  sender.isEnabled = false
  let harfString = sender.title(for: .normal)!
  let harf = Character(harfString.lowercased())
}
```

3- "Oyun.swift" dosyasını açın.

4- "Oyun" struct'ının içerisine **"harfTahminleri"** adında, Character dizisi türünden bir değişken tanımlayın.

5- "Oyun" struct'ının içerisinde, **"tahminYapildi"** adında bir mutating metod tanımlayın. Bu metod, parametre olarak bir Character nesnesi almalı.

6- "tahminYapildi" metodunun içinde, parametre olarak alınan harfi **"harfTahminleri"** dizisine ekleyin.

7- "tahminYapildi" metodunun içinde, eğer **"kelime"** değişkeni alınan harfi içermiyorsa, **"kalanTahminSayisi"** değişkenini 1 azaltın.

8- "yeniTur" metodunuzun içerisinde, **"simdikiOyun"** değişkeninin tanımına, **"harfTahminleri"** parametresine boş bir dizi verin.

9- "butonTiklandi" metodunuzun içerisinde, **"arayuzuGuncelle"** metodundan önce **"harf"** değişkeni ile **"simdikiOyun"** nesnesi üzerinden **"tahminYapildi"** metodunu çağırın.

Uygulamanızı çalıştırın. Her bir butona dokunduğunuzda, o butondaki harfin **"harfTahminleri"** dizisine eklendiğini ve **"kalanTahminSayisi"** değişkeninin her yanlış harf için 1 eksildiğini göreceksiniz.

Adım 5: Gösterilen Kelimeyi Oluşturma

Artık **"kelime"** ve **"harfTahminleri"** değişkenlerini kullanarak, kelimenin tahmin edilmeyen harfleri bulunmayan bir halini oluşturabilirsiniz. Örneğin, kelime "büyüleyici" ise ve oyuncu "b", "d", "i", "ü" harflerini tahmin ettiyse, kelime ekranın altında "b ü _ ü _ _ _ i _ i" şeklinde görünmelidir.

1- "Oyun" struct'ını içerisine **"formatlanmisKelime"** adında bir computed property ekleyin. Bu özelliğin değeri, şöyle hesaplanmalı:

- **"kelimeTahmini"** isimli boş bir String değişkeni ile başlayın.
- Döngü ile **“kelime”** değişkenindeki her karakterinin üzerinden geçin:
    - Eğer karakter **"harfTahminleri"** değişkeninin içinde yer alıyorsa, onu bir String'e dönüştürün ve **"kelimeTahmini"** değişkenine ekleyin. Öbür türlü **"kelimeTahmini"** değişkenine "_" karakterini ekleyin.

2- "arayuzuGuncelle" metodunda **"cozumLabel"** elemanının "text" özelliğini **"formatlanmisKelime"** değişkeninin değeri olacak şekilde güncelleyin.

Bir sorun fark etmiş olabilirsiniz: birden fazla "_" karakteri yan yana gelince, tek bir çizgi gibi gözüküyor. Bunu düzeltmek için, arayüzü güncellerken karakterlerin arasında birer boşluk ekleyin:

3- "arayuzuGuncelle" metodunun ilk satırında, **"harfler"** isimli, String dizisi türünden bir değişken tanımlayın.

4- "formatlanmisKelime" değişkeninin her karakterinin üzerinden geçerek karakteri String'e dönüştüren ve **"harfler"** dizisine ekleyen bir for döngüsü tanımlayın.

5- "boslukluKelime" adında bir sabit tanımlayın. Bu sabitin değerini, **"harfler"** dizisinin `joined(separator: )` özelliği olarak belirleyin. "separator" parametresine " " yani boşluk String'ini verin.

6- "cozumLabel" elemanının "text" özelliğini **"boslukluKelime"** değişkeninin değeri olacak şekilde güncelleyin.

Uygulamanızı çalıştırın. Her harfin arasında bir boşluk görüyor olmalısınız.

Adım 6: Kazanma ve Kaybetme Durumlarını Algılama

1- "ViewController" sınıfının içerisinde **"oyunuGuncelle"** adında parametresiz bir fonksiyon oluşturun.

2- "butonTiklandi" metodunun en sonunda **"oyunuGuncelle"** fonksiyonunu çağırın.

3- Şimdi sıra, **"oyunuGuncelle"** metodunu yazmaya geldi. Bu metod, şöyle çalışmalı:

- Eğer **"kalanTahminSayisi"** değeri sıfıra eşitse, **"toplamYanlis"** değişkenine 1 ekleyin.
- Eğer **"kelime"** değişkeni **"formatlanmisKelime"** değişkenine eşitse, **"toplamDogru"** değişkenine 1 ekleyin.
- Hiçbir koşul doğru değilse, arayüzü güncelleyin.

4- "toplamDogru" ve **"toplamYanlis"** değişkenlerini, yeni turu başlatacak bir didSet ekleyin.

Uygulamanızı çalıştırın. Kazanılan ve kaybedilen turların doğru sayılıp sayılmadığını kontrol edin.

Adım 7: Butonları Yeniden Çalıştırma ve Oyunu Bitirme

1- "harfButonlariniAktiflestir" isimli bir metod tanımlayın. Bu metod, **"aktif"** isimli Bool türünden bir parametre alsın.

2- "harfButonlariniAktiflestir" metodunun içinde, döngü ile her butonun "isEnabled" özelliğini "aktif" parametresine eşitleyin.

ÖRNEK

```
func harfButonlariniAktiflestir(_ aktif: Bool) {
  for buton in harfButonlari {
    buton.isEnabled = aktif
  }
}
```

3- "yeniTur" metodunu, şu biçimde güncelleyin:

- Eğer **"kelimeListesi"** dizisi boş değilse, metodda daha önce yapılanları yapın. **"harfButonlariniAktiflestir"** metodu ile butonları kullanılabilir hale getirin.
- Eğer kelime dizi boşsa, **"harfButonlariniAktiflestir"** metodu ile butonları kullanılamaz hale getirin.

Uygulamayı çalıştırın. Kelime kalmaya ve butonlar kullanılamaz hale gelene kadar adam asmaca oyununu oynayabiliyor olmalısınız! 🎊