TEMEL ARAYÜZ OLUŞTURMA (1)

- **Xcode'a Giriş**
  - **Proje Oluşturma**
  - **Proje Çalıştırma**
- **Kaynak Kod İncelemesi**
  - **AppDelegate.swift**
  - **ViewController.swift**
  - **Main.storyboard**
- **Temel Arayüz Tasarımı**

XCode

Xcode, iOS işletim sistemine sahip cihazlara uygulama oluşturmak için gerekli olan tüm öğelere sahiptir. Uygulama oluşturmak için gerekli olan sayfa ve kaynakları düzenler. Hem kod hem de kullanıcı arabirimleri için düzenleyici sağlar. Ayrıca geliştirilen uygulamalar için simülatör sağlar.

1. Proje Oluşturma

1. Proje teması olarak iOS ve Single View Application seçilecektir.
2. Proje bilgileri şu şekilde girilecektir:
3. **Product Name:** Projenize isim verebilirsiniz.

   Team: Ekip olarak yapılan projelerde kullanıcı eklenebilir. Boş bırakabilirsiniz.

   Organization Name: Organizasyon veya kendi isminizi verebilirsiniz.

   Organization Identifier: Organizasyon kimliği olarak girilir. Değeri ktu olarak girebilirsiniz.

   Bundle Identifier: Bu kimlik otomatik olarak oluşturulacaktır.

   Language: Swift

   Devices: Universal (Universal uygulamalar hem iPhone hem de iPad cihazlarında çalışır.)

   Use Core Data: Seçili değil.

   Include Unit Tests: Seçili değil.

   Include UI Tests: Seçili değil.
4. Bir sonraki adımda çalışma alanı belirlenir. Create Git repository on seçeneğini seçili değil olarak belirleyiniz.
5. Çalışma alanı penceresinde bir geliştirme ekibi gerektirdiğini bildiren hata simgesi görebilirsiniz. Bu uyarı, henüz iOS geliştirme için Xcode ayarlamadığınız anlamına

gelir. Ancak uygulamayı simülatörde çalıştırmak için bir geliştirme ekibine ihtiyacınız yoktur.

2. Uygulama Çalıştırma

Proje oluştururken şablon olarak **Single View Application** olarak seçildiğinden dolayı herhangi bir kod yazmadan uygulama çalıştırılabilir. Uygulamayı çalıştırmak için XCode'da bulunan **iOS Simulator** kullanılır. Simülatör sayesinde her türlü boyut ve çözünürlükte tasarım yapılabilir. Bu ders için genellikle **iPhone 7** seçeneğini kullanacağız.

1. Xcode araç çubuğundaki **Scheme pop-up** menüsünden **iPhone 7** seçiniz.
2. Xcode araç çubuğunun sol üst köşesinde bulunan **Run** düğmesini tıklayınız.

Xcode, projenizi oluşturmayı tamamladıktan sonra, simülatör otomatik olarak başlar. İlk kez başlamak biraz zaman alabilir.

Kaynak Kod İncelemesi

1. AppDelegate.swift

AppDelegate.swift kaynak dosyasının iki temel işlevi vardır:

- **AppDelegate** sınıfınızı tanımlar. **AppDelegate** sınıfı, uygulamanızın içeriğinin çizildiği pencereyi oluşturur ve uygulama içindeki durum geçişlerine yanıt vermek için bir yer sağlar.
- Uygulamanızın giriş noktası ve giriş olaylarını uygulamanıza ileten bir çalışma döngüsü oluşturur. Bu çalışma, dosyanın üst kısmında görünen **UIApplicationMain** özelliği (@UIApplicationMain) tarafından yapılır.

UIApplicationMain özelliğini kullanmak; **UIApplicationMain** fonksiyonu çağırıp **AppDelegate** sınıfının adını temsilci sınıfının adı olarak belirlemektir. Bu işlem ile sistem bir uygulama nesnesi oluşturur. Uygulama nesnesi, uygulamanın yaşam döngüsü yönetiminden sorumludur. Sistem ayrıca **AppDelegate** sınıfının bir örneğini oluşturur ve onu uygulama nesnesine atar. Bu şekilde sistem uygulamanızı başlatmış olur.

AppDelegate sınıfı, yeni bir proje oluşturduğunuzda otomatik olarak oluşturulur. Oldukça alışılmadık bir şey yapmadığınız sürece, uygulamanızı başlatmak ve uygulama düzeyindeki etkinliklere yanıt vermek için Xcode tarafından sağlanan bu sınıfı kullanmalısınız. **AppDelegate** sınıfı, **UIApplicationDelegate** protokolünü uygular. Bu protokol, uygulamanızı kurmak, uygulamanın durum değişikliklerine yanıt vermek ve diğer uygulama düzeyindeki etkinlikleri gerçekleştirmek için kullandığınız bir takım fonksiyonları tanımlar.

AppDelegate sınıfı tek bir değişken içerir: **window**.

```
var window: UIWindow?
```

Bu değişken, uygulama penceresine yapılan bir referansı saklar. Bu pencere, uygulamanızın görünüm hiyerarşisinin kökünü (root) temsil eder. Tüm uygulama içeriğinizin çizildiği yerdir.

Pencere özelliğinin isteğe bağlı olduğunu unutmayın; bu, bir noktada hiçbir değeri olmayabilir (null olabilir) anlamına gelir.

AppDelegate sınıfı ayrıca aşağıdaki fonksiyonları içerir:

```
func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplicationLaunchOptionsKey: Any]?) -> Bool
func applicationWillResignActive(_ application: UIApplication)
func applicationDidEnterBackground(_ application: UIApplication)
func applicationWillEnterForeground(_ application: UIApplication)
func applicationDidBecomeActive(_ application: UIApplication)
func applicationWillTerminate(_ application: UIApplication)
```

Bu fonksiyonlar, uygulama nesnesinin uygulama temsilcisi ile iletişim kurmasına izin verir. Uygulama durumunu değiştirirken (örneğin, uygulama başlatma, arka plana geçiş yapma ve uygulama sonlandırma) uygulama nesnesi, karşılık gelen temsilci fonksiyonu çağırarak uygulamanıza yanıt verme fırsatı verir. Bu fonksiyonların doğru zamanda çağrıldığından emin olmak için özel bir şey yapmanıza gerek yoktur, uygulama nesnesi bu işi sizin yerinize yapar.

Bu derste, herhangi bir özel uygulama temsilci (custom app delegate) kodu kullanmayacağınızdan, AppDelegate.swift dosyasında herhangi bir değişiklik yapmanız gerekmez.

2. ViewController.swift

ViewController sınıfı, **UIViewController'ın ViewController** adlı özel bir alt sınıfını tanımlar. İlk anda, bu sınıf sadece **UIViewController** tarafından tanımlanan fonksiyonları alır. Bu fonksiyonları düzenlemek için **UIViewController'da** tanımlanan fonksiyonları **override** etmek gerekir.

ViewController.swift dosyasında görebileceğiniz gibi, template uygulama hem viewDidLoad() hem de didReceiveMemoryWarning() yöntemlerini override eder.

3. Main.storyboard

Uygulamanızı iPhone 7 Simulator uygulamasında çalıştırdığınızda, bu sahnedeki görüntü, cihaz ekranında gördüğünüz şeydir. Bununla birlikte, kanvastaki sahne, simülatörün ekranıyla aynı boyutlara sahip olmayabilir. Tuvalin alt kısmında ekran boyutu ve yönünü seçebilirsiniz.

> Kanvas belirli bir cihaz ve yönlendirme gösterirken, adaptif bir arabirim oluşturmak önemlidir; bu, herhangi bir cihazda ve herhangi bir yönünde iyi görünmesi için otomatik olarak ayarlanan bir arabirimdir. Arayüzünüzü geliştirirken, arayüzünüzün farklı boyut ekranlarına nasıl uyum sağladığını görmenizi sağlayan tuval görünümünü değiştirebilirsiniz.

Temel Arayüz Tasarımı

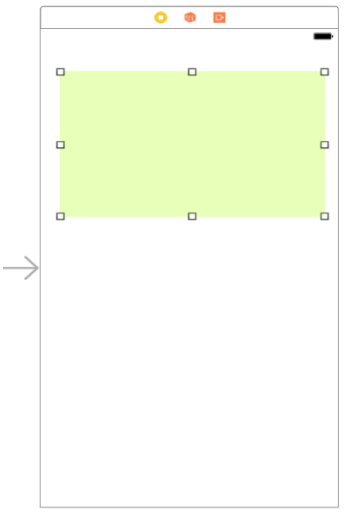
Xcode, storyboard dosyasına ekleyebileceğiniz bir nesne kütüphanesi (**Object Library**) sağlar. Bunların bazıları butonlar ve metin alanları gibi kullanıcı arayüzünde görünen öğelerdir.

Kullanıcı arabiriminde görünen öğelere view adı verilir. Viewler, kendilerini ekranda görüntüleme ve kullanıcı girdisine tepki gösterme de dâhil olmak üzere çeşitli davranışlara sahiptir.

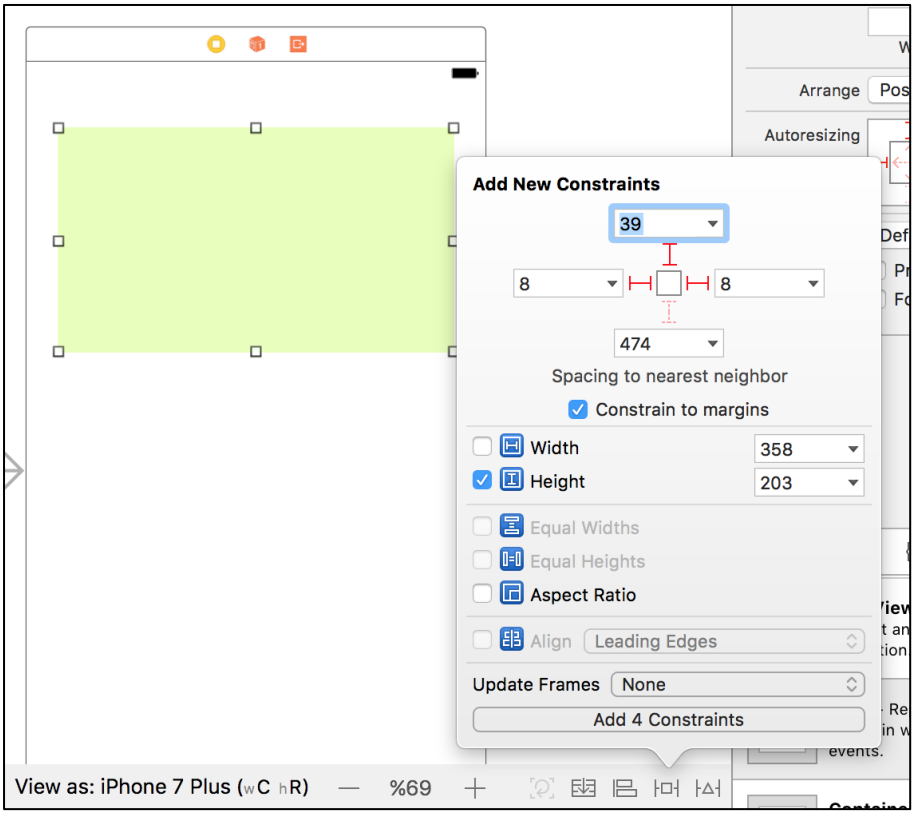
View'lerin yerleşimi/düzeni (**layout**) ve bu düzenin farklı cihazlarda aynı görüntüyü vermesi uygulama geliştirirken dikkat edilmesi gereken en önemli konudur. Bunun için Swift kullanıcılara **AutoLayout** özelliği ile yerleşim düzenini kolayca tasarlayabilecekleri bir ortam sunmaktadır.

Şimdi AutoLayout özelliğini kullanarak bir ara yüz tasarımı yapalım.

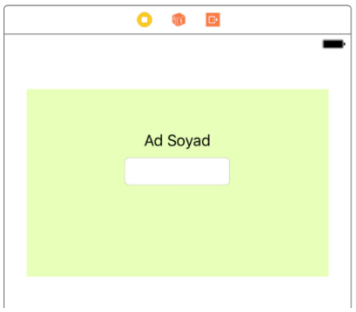
1. **Main.storyboard** tıklayın ve ekranda bulunan view üzerine sağ alt köşede bulunan nesne kütüphanesinden (**Object Library**) yeni bir **View** ekleyin. Yandaki gibi arka plan rengini özellik denetçisinden (**Attributes Inspector**) değiştirin.



2. View seçiliyken sağ alt köşedeki Add New Constraints butonuna tıklayın ve açılan pencereden sağ ve sol genişlikler aynı olacak şekilde değerleri düzenleyin. Kenarlara olan uzaklığı temsil eden bu değerlerin yanında bulunan kesik kırmızı çizgilere tıklayarak layout için sabitleri (constraints) belirleyin. View'in ekranın altına olan uzaklığını belirleyen kesikli çizgiye tıklamayın. Bunun yerine şekildeki gibi Height'i işaretleyin. Böylece uygulamanız farklı cihazlarda kullanılırken üst, sağ ve sol kenarlara olan uzaklığı belirlediğimiz sabitlere göre (8,39,8) ayarlanırken, alt kenara olan uzaklığı view'in yüksekliğine (474) göre otomatik belirlenecektir. Add 4 Constraints yazan butona tıklayarak yaptığınız seçimleri constraints olarak belirleyin.



3. Ekranın altındaki View as:İPhone 7Plus yazan çerçeveye tıklayın ve açılan bölümden uygulamanızın farklı cihazlarda ve dikey/yatay görünümdeki davranışını inceleyin.
4. View'ın üzerine resimdeki gibi bir Label ve TextField ekleyin.



5. Label ve TextField'i birlikte seçin ve ölçü denetçisi (Size Inspector) penceresinden Arrange açılır kutusuna tıklayıp "Center Horizontally In Container" (Taşıyıcının yatay ortası) seçin.
6. Label'ı seçin ve Add New Constraints penceresinden pencerenin üstüne olan uzaklığı gösteren kesikli çizgiye tıklayın. Ardından genişliğin sabit kalması için Width'i seçip aktifleştirin ve seçiminizi constraints olarak belirlediğiniz butona basın.
7. Constraint ekleme butonunun solundaki Align butonuna tıklayın. Açılan pencereden "Horizontally in Container" seçeneğini seçiniz ve alttaki butonla seçiminizi constraint olarak ekleyin.
8. Farklı cihazları ve yatay/dikey görünümü seçerek uygulamanızın nasıl göründüğünü kontrol edinin.
9. **TextField'ın altına GÖSTER yazan bir buton ekleyin ve layout ayarını yapın.**